PERSONALNI
računari
79
CENA 300 D.
FEBRUAR
1992.

SOFTVER

WORD PERFERCT FOR WINDOWS
AUTO SHADE 2
MICROSOFT PUBLISHER
KING JAGUAR
GAUSS
qCAD

SPECIJALNI DODATAK NA 32 STRANE

ŠKOLA CLIPPERA
WINDOWS NT

SKENERI
HP SCANJET II

ŠTAMPAČI
HP DESKJET 500

RAČUNARI
COMPAQ
HP 95 XL

VIDEO KARTICE



IBIS SYS
ekonomski biro

ima bolju ideju. . .

POZIVAMO

— sve korisnike PC-računara nezadovoljne servisnom podrškom svojih dobavljača,

— sve distributere i dealere PC-računara, koji imaju teškoća u organizovanju sopstvenog servisa u garantnom i vangarantnom roku,

NA DOGOVOR

Tehnička laboratorija

IBIS SYS
SOFTWARE HARDWARE INŽENJERING

servisira
dograduje
zamenjuje **RAČUNARE**
povezuje

Telefon tehničke laboratorije: 011/609-650, 605-890

PERSONALNI
računari 79

časopis za informatiku i računarstvo
YU ISSN 0352-7221
Izdaje BIGZ
februar 1992. cena 300 D.

Izdaje i štampa
Beogradski izdavačko-grafički zavod
11000 Beograd
Bulevar Vojvode Mišića 17
Generalni direktor
Ilija Rapajić
Direktor Novinskog sektora
Zoran Nikodijević
Glavni i odgovorni urednik
Jovan Regasek
Zamenik gl. i odg. urednika
Esad Jakupović
Tehnički urednik / sekretar redakcije
Natalija Desančić
Urednik za hardver
Dejan V. Veselinović
Stručne redakcije
Vesna Cosić (aktuelnosti), Nenad Batočanin (baze podataka), Miodrag Lovrić (radne tabele), Dejan Ristanović (softver), Jovan Skuljan (programiranje), Dejan Veselinović (hardver), Dragutin Vuković (mreže), Zoran Životić (softver)
Adresa redakcije
11000 Beograd
Bulevar vojvode Mišića 17/III
Telefoni
653-743 (redakcija)
650-528 (prodaja)
Agencija BIGZ-a
(011) 651-793 653-585
Teleks
11855 BIGZ YU
Telefaks
(011) 648-140 647-955
Pretplata za zemlju
Za osam meseci 137$ (11 brojeva)
Za jednu godinu 2750 (11 brojeva)
(na žiro-račun D.P. BIGZ 60602-603-23264)
Pretplata za inostranstvo
Za jednu godinu 51 USD, 80 DEM, 67 CHS, 27 GBP, 271 FRF
(na devizni račun D.P. BIGZ 60811-620-16101-820701-999-03377)
Zbog usporenog prenošenja uplata preko banke, molimo pretplatnike da nam posle svake nove uplate odmah pošalju foto-kopiju uplatnice
Rukopisi se ne vraćaju.

SEZAM
sistem za modemske komunikacije
(011) 648-899 24 časa dnevno
Upravnik sistema
Jovan Regasek
Administratori na sistemu
Zoran Životić i Dejan Ristanović
Moderatori
Dejan Bašinac (CIVILIZACIJA) Jovan Bulajić (PC USER), Aleksandar Damnjanović (PC OS) Slobodan Kalezić (ORKA) Zoran Kehler (MOST), Vladimir Kostić (PC SOFT), Vladimir Kratohvil (VICEVI) Esad Jakupović (RAČUNARI), Dalibor Lanik (ATARI ST), Pavle Pavković (PC SOFT) Dejan Ristanović (NOVOSTI I KOMUNIKACIJE) Bojan Tepavčević (FORUM) Dejan Veselinović (PC HARD) i Zoran Životić (SEZAM)
Sekretarica sistema
Natalija Desančić
Softver
Zoran Životić (glavna aplikacija) i Dejan Ristanović (eksterni programi)
Hardver
Server: COMPAQ DESKPRO 386/25 3 MB RAM a tvrdi disk 300 MB (COMPAQ Minhen) Mreža: dve stanice COMPAQ 286N/12, 1 MB RAM-a (COMPAQ Minhen) tri stanice „Ultra" AT 12 MHz, D-LINK DE 100 LanSmart ("Ultra" Skopje). Modemi ViVa (Thelema Communitalena Beograd)

Na osnovu Mišljenja Sekretarijata za informacije SR Srbije broj 413-01-46 od 4. 2. 1991. na ovaj časopis plaća se osnovni porez na promet po povlašćenoj tarifi od 3 odsto

SADRŽAJ

4 Vesti
ŠTA IMA NOVO
8 Procesori / F8680
PC U JEDNOM ČIPU
9 Računari / HP 95XL
NAJMANJI NA SVETU
10 Štampači / HP DJ 500C
U SVIM BOJAMA DUGE
11 Skeneri / HP IIC
UJEDINJENE BOJE PAKARDA
14 Uporedni test
PATULJCI I DŽINOVI
18 Video kartice
OBRAČUN KDD OK PIKSELA
20 Disk kontroleri
HARDVERSKI KEŠ
22 Standardi / Unicode
ZA SVAKI JEZIK NA SVETU
23 Štenderdi / SCSI 2
ŽIVOT NA SABIRNICI
26 Tržište
SOFTVERSKA TRKA VEKA
28 Operativni sistemi / Windows NT
WINDOWS ZA VISOKU KLASU
32 CAD / AutoShede 2
SVETLOSTI I SENKE
34 Obreda teksta / WP4WIN
DALEKO OD PERFEKTNOG
38 Stono izdevaštvo / MS Publisher
VENTURA ZA SIROMAŠNE
40 Kompejleri / King Jaguar
UBRZANE TABELE
42 Programski jezici / Gauss 2.0
ZA LJUBITELJE MATRICA
44 Kriptografija / DES
KOD IZ ĐAVOLJEG SEMENA
46 CAD / NURBS
UMETNOST MODELIRANJA
48 CAD / qCAD
RADOST PROJEKTOVANJA
51 Tehnike progremirenja / C
ŠKOLSKI RAM DISK
54 Tehnike programiranje / Windows
U SREDIŠTU SISTEMA



Suočena sa delikatnim izborom između malog i velikog računara, maneken Olja Todorović (agencija "Click") bez razmišljanja se odlučila za mali, na blago iznenađenje ekipe za naslovnu stranu "Računara" – fotografije Nenad Marjanović, make up Darija Miloš, kreacije modela Marijana Ćetković. Nije, međutim, reč o ženskom hiru, niti o činjenici da male stvari prirodno više imponuje ženskom ukusu. Ova "mala, malecka, ovoliđna" mašina jeste najmanji na svetu, ali ipak pravi personalni računar XT klase – "HP 95XL".

56 Tehnike programiranja
RASTAVI ME NEŽNO
61 Tehnike programiranja / C
DINAMIČKA POLJA
62 Dejanove pitalice
NAGRADNA PREMETALJKA
64 Savetnik
TURBO PASCAL
66 Zoren Životić
MDJA ŠKOLA C-a
65 Savetnik
CLIPPER
66 Savetnik
LOTUS 1-2-3
69 Dejan Ristanović
BAJTOVI LIČNE PRIRODE
71 Sezem
SEZAMFILE
SEZAM BILTEN

martovski broj časopisa
računari
izlazi iz štampe sredinom marta 1992.

Procesori

MONSTRUM U ČIPU

Poslednjih dvogodišnjeg relativno mirnog perioda pred nama je prilično vatreno proleće na tržištu mikroprocesora. Sve više je novih procesora koji čine prelaz iz 32-bitnog u 64 bitni svet. U novoj generaciji 50 MHz RISC procesora prvi je na tržište stigao T9000 iz konačno probudjenog Inmosa, a u stopu ga slede Intel 860XP i MIPS R4000.

Jedan od procesora koji svakako najviše obećavaju je „Texas Instrumentsov" SuperSparc nazvan još i „Viking". Novi RISC monstrum na preko tri miliona tranzistora integrišu dva paralelna celobrojna procesora, brzi vektorski procesor, posebne MMU za naredbe i podatke, kao i ukupno 32 K keša (po 16 K za instrukcije i podatke sa paralelnim pristupom) uz 64-bitnu spoljnu sabirnicu propusnosti do 400 megabajta/s. Sva to radi na 50 MHz podržava keširanje drugog nivoa do 1 MB i poslužje „znatno preko 50 VAX MIPS". Novi procesor je, naravno, u potpunosti kompatibilan sa svim drugim SPARC procesorima a „Texas" je najavio i kompletan čipset za ekspresno dizajniranje radnih stanica oko novog procesora. Procesori koji bi u trenutku dok ovo čitate trebali biti lansirani su Motorola 88110, HP „Snake" RISC i IBM POWER II u jednom čipu. Možda i Intelov 586 koji kasni ll kasni...

Da bi barem delimično nadoknadio kašnjenje, „Intel" je u međuvremenu pored konačno ispravne verzija 50 MHz 486, koja se na modulu sa 82485 brzim keš kontrolerom i 256 Kb write-back keša naziva APOGEE predstavio i posebnu verziju 33 MHz 486 sa internim lakiduplikatorom. Kod novog P24 Turbo486 sva unutrašnjost procesora (CPU FPU MMU keš) radi na 66 MHz, dok veza sa spoljnim svetom radi na 33 MHz. Praktično ubrzanje koje se da postići zamenom postojećeg 33 MHz 486 sa 66/33 MHz P24 bi trebalo biti između 50% i 80% – možda i isplativije nego skakati na čisti 50 MHz 486 sa znatno skupljim keš memorijama i RAM-om. *(N N)*

Portabli

486/50 PORTABL



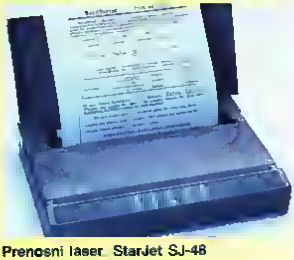
Prenosni džin: Dolch P.A.C. 486-50E

Ako odlučite da kupite „Dolch P.A.C. 486-50E", svakako mu obezbedite neki ozbiljan posao – reč je o prevozivom računaru sa 486DX procesorom koji radi na 50 MHz, 8 MB RAM-a (proširivo do 32 MB) i 200-MB SCSI hard diskom sa keš kontrolerom. Tastature predstavlja mešavinu klasične IBM AT tastature sa 84 tastera i proširenog modela, a monitor je monohromatski gas-plazma, dok je za one koji vrše vola kolor obezbedena alternativa u vidu Hitačijevog kolor LCD-a sa aktivnom matricom na bazi tankoslojnih tranzistora. Za korisnike kojima ni to nije dovoljno predvidena je mogućnost ugradnje 128 KB keš memorije, a tu su i osam SIMM utičnica, tri ISA i tri EISA slota, od čega jeden EISA slot drži SCSI kontroler sa 4 MB keša, u drugom se nalazi 16-bitna VGA kartica sa 1 MB video RAM-a a u trećoj je smeštene I/O kartica. Sve to se napaja iz izvora jačine 225 W. Cena sasvim odgovara klasi i iznosi 16 995.

Kontakt adresa: Dolch Computer Systems, 372 Turquoise St, Milpitas, CA 95035; tel. 800-538 7506, 408-957-6575; fax 408-263-6305

Štampači

VIŠENAMENSKI PORTABL ŠTAMPAČ



Prenosni laser: StarJet SJ-48

Sa 24 KB memorije i opcionim NiCd baterijama „StarJet SJ-48" višenamenski ink-jet printer štampa do 40 stranica bez punjenja. Glave za štampanje se 64 mlaznice obezbeduju laserski kvalitet uz buku od samo 45 dB. SJ-48 je opremljen se četiri rezidentna bit-mapirana fonta, dve 1.2 MB diskala sa praćećim programima i fontovima za DTP, kao i drajverima za *Windows 3.0*. Maksimalna rezolucije iznosi 360 dpi, uz maksimalnu brzinu od 6.5 strana u minuti pn 20 cpi, a na raspolaganju je i tzv. *quarter mode* koji omogućava štampanje 273 u redu pn 20 cpi. Cena $499 + $50 za akumulator.

Kontakt adresa: Star Micronics America Inc., 420 Lexington Ave., Suite 2702, New York, NY 10170; tel. 212-986-6770; fax 212-286-9063

Notebook

"COMPANION PC" ZAJEDNIČKIM SNAGAMA

Veći od džepnog računara, lakši od notebook računara, prodaje se brže od Psion Seris 3 ili HP 95LX – to je cilj koji su sebi postavili projektanti „Companion PC"-a. A reč je o kombinaciji proizvoda „Microsofta", „Lotusa", „Phoenixa", „Intela" i „Chips & Technologiesa". Companion PC će raditi na obične AA baterije i koštaće manje od $1 000 dolara. Za razliku od notebook-a biće toliko lagan da korisnik neće uopšte morati da razmišlja da li da ga nosi se soodm ili ne – zajedno sa punom lastaturom, biće težak svega oko 900 grama! Za nešto više para, kupac će moći da dobije i 1.8" hard disk, naravno ukoliko se snabdevači budu držali plana.

"Phoenix", „Lotus" i „Microsoft" su pristupili kooperativnom programu za podršku proizvođačima „Companion PC"-a. „Phoenix" će obezbediti hardverske i softverske reference i BIOS, „Microsoft" deje licencu za DOS 5.0 u ROM-u, a „Lotus" ROM-baznice poslovne aplikacije. Sa svim ovim komponentama, proizvođačima neće ostati previše posla – glavni problem će biti pronalaženje kupaca.

Trendovi

MULTIPROCESORSKI 486-50

Dok 586 ne stigne, nova moda su višeprocesorski 486 sistemi. Uzmite nekoliko (do 4) Intelovih 50 MHz Apogee modula (svaki sadrži 50 MHz 486, 82485 kes-kontroler i 256 Kb ultrabrzog write-back keša sa hladnjakom), vežite ih preko brze 64 bitne sabirnice sa zajedničkom memorijom i Multiprocessing Interrupt Controllerom, uz obavezni EISA bus za I/O. RAM mora biti sa ECC popravkom grešaka i naravno, preplitenjem, uz gornju granicu kapaciteta najmanje 64 MB. Na tom principu su građeni Micronics Mpro SMP-2000 (2 procesora) i SMP-4000 (4 procesora). Podrška SCO Unix MPX se podrazumeva. Početne cene su ispod 10 000 USD za ogoljene verzije sistema.

NEC-OV KOLOR PORTABL

NEC ProSpeed 486SX/C je jedan od prvih računara ove klase koji koristi „Intelov" procesor 486SX. Opremljen je lakode NEC-ovim kolor ekranom sa aktivnom matricom na bazi tankoslojnih tranzistora (Thin Film Transistor) koji istovremeno prikazuje 256 boja iz palete od 4096 u VGA rezoluciji sa uglom gledanja od 180 stepeni uz mogućnost istovremenog korišćenja i eksternog monitora. ProSpeed 486SX/C radi na 20 MHz i opremljen je sa 2 MB RAM-a (proširivo do 20 MB u obliku SIMM-ova), ugrađenim 32 bitnim EISA ekspanzionim slotom pune dužine, 3.5" flopijem, hard diskom od 120 MB i MS DOS-om 5.0. Od ostalih mogućnosti obično rezervisanih za stone računare tu su još tri 16-bitna ekspanzione slote pune dužine, jedan poludužinski 8 bitni slot i mesto za dva poluvisinska drajva. Dimenzije ovog velikog malog računara su 376 x 396 x 107 mm, a težina iznosi 7.62 kg, sve po cani od oko 6 000 GBP.



Prvi 486SX laptop sa 256 boja. Nec ProSpeed 486SX/C

Kontakt adresa: NEC (U.K.) Ltd., NEC House, 1 victoria Rd., London W3 6UL, U.K.; tel. 44-81 993 8111, fax 44-81-992 7161

Mylex Multiprocessing System kao dodatnu prednost ima i Disk Array Subsystem u posebnom kućištu. Mylex EISA RAID kontroler za paralelno vezane diskove je pravi biser: 66 MIPS i960CA-33 superskalarni RISC sa ultra-bizim DMA pogonii 5 (de, pet) paralelnih NCR 53C720 Fast Wide SCSI-2 procesora svaki sa protočnošću do 20 megabajta/s. Četiri se vezuju na diskove za podatke (po 16 bita 64-bitne reči) a peti na redundantni disk koji je u stanju da ukoliko se ceo jedan disk u sistemu uništi u minut dva na novom disku obnovi sve podatke. Hot Replacement odlika omogućuje da se zamena vrši dok sistem i dalje neometano radi! Naravno, da ne zaboravimo, na kartici ima mesta za do 64 megabajta write-back disk keša. Za cenu bolje ne pitajte.

Fotografija

PC KAO MRAČNA KOMORA

Foto emajleri koji imaju PC više ne moraju da čekaju na razvijanje filmova i izradu slika, da kupuju filmove niti da slažu slike u albume. Novi „Logitechov" uređaj veličine instant-fotoaparata nazvan „FotoMen" može da snimi 32 slike u jednoj sesiji opremljen je automatskim



Elektronski foto-aparat Logitech FotoMan

fiksnim neograničenim fokusom, filterima za snimanje u eksterijeru i ugrađenim NI-Cd baterijama. Po završenom snimanju, kamera se jednostavno priključi na serijski port PC-a i slike za nekoliko sekundi prenesu u program za obradu slika *FotoTouch*. Program radi pod *Windows* okruženjem, prikazuje slike u 256 nijansi sivog i omogućava uklanjanje pozadine, pravljenje izreza, skaliranje cele slike, podešavanje kontrasta i osvetljenja, kao i nekoliko specijalnih efekata. Tako obrađene slike se mogu uneti u bilo koju aplikaciju a prilikom snimanja na disk se kompnimuju tako da zauzmu što manje mesta.

"FotoMen" predstavlja kompletan paket opremljen svim potrebnim stvarima, uključujući i doživotnu garanciju i podršku, sve po ceni od $799.

Operativni sistemi

STIŽE OS/2 2.0

IBM za mart najavljuje pojavu operativnog sistema za devedesete. OS/2 2.0. Okružene će raspolagati *Workplace Shell*-om i podržavaće DOS *Windows* i 16-bitne OS/2 aplikacije. Pri čemu će se sve zajedno moći da rade na istoj platformi. Logično pitanje koje se nameće na ovakvu najavu glasi: *Pa zar nije IBM obećao isti taj OS/2 2.0 za kraj 1991?* IBM bi odgovorio da jeste, a činjenica je da sistem i pored zakašnjenja neće raspolagati svim funkcijama koje su bile najavljene i demonstrirane na raznim sajmovima, kao što je mogućnost primene cut&paste operacija među *Windows* aplikacijama. Verzija koja je planirana za prolećni decembar će podržavati DOS i sadržeće *Workplace Shell* ali će podržavati samo full-screen *Windows* aplikacije, a zvanično „opravdanje" IBM ovih predstavnika glasi da je „određen broj kupaca izjavio kako se *Windows* ne uklapa u njihove planove, već žele OS/2 odmah."

Procesori

IBM + APPLE + MOTOROLA = PowerPC

Dok „Intel" priprema (i uveliko kasni sa) 586, „Apple", IBM i „Motorola" (da li ste ikada ova tri imena mogli da zamislite zajedno?) za kraj ove godine pripremaju prvu seriju PowerPC procesora. Reč je o implementaciji snažne IBM POWER RISC arhitekture u samo jednom čipu. Da bi se skalamerija snažnija i od Intelovog 860XP sabila u jedan čip potrebno je kombinovati najbolje iz „Motorolne" i IBM-ove 0.5-mikrometarske CMOS tehnologije. Za 1993. najavljuje se model u jednom čipu koji će bitno nadmašiti 100 VAX MIPS.

Uz novu organizaciju PowerOpen koja će definisati standarde kompatibilnosti za novi procesor najavljene je i solidne početna podrška *Mac OS/2* i *Unix OSF/Motif* operativnu sistemi sa ogromnom gomilom softvera.

Procesori

IBM + INTEL = ???

IBM, međutim ne zaboravlja na „Intel". Novi ugovori omogućavaju IBM da sam za sebe proizvodi 486 procesore s tim što ih po potrebi može modifikovati i unapređivati. Takođe IBM ima pravo da primenjuje nove verzije „Intel" procesora pre svih ostalih. Za uzvrat „Intel" će takođe moći da prodaje IBM-unapređene verzije svojih procesora drugima. Sve ovo pokazuje da 386 arhitektura ipak gubi dah i da mnogo više od onoga što je učinjeno sa 486 i ne može biti postignuto. Razlozi su znani: mnogo leže je paralelizovati i ubrzavati procesore sa kompleksnim skupovima naredbi pogotovu asimetričnim i tradicionalno osmišljenim (kao u 386/486) nego one sa RISC arhitekturom. Sveže ideje IBM-ovih iskusnih inženjera će zato dobro doći.

MS ALPHABETPLUS 1.0

Microsoft je nedavno objavio puštanje u prodaju paketa *AlphabetPlus, v1.0* – reč je o potpuno novom proizvodu čiji je glavni cilj obezbeđivanje podrške za korišćenje različitih jezika istočne i srednje Evrope u Microsoft-ovim DOS-orijentisanim paketima *Word Multiplan, Chart and Project* i *Works*. *AlphabetPlus* omogućava kucanje, prikazivanje, snimanje i štampanje na čak naest jezika uključujući i „ćirilične" jezike a u paketu se nalazi i veliki broj drajvera i fontova za štampače, *User's Guide* na sedam jezika odgovarajuće nelepnice za tastaturu za sve podržane jezike i još dosta drugih važnih sitnica, kao što je program za konverziju leksikalnih fajlova iz neslandardnih fontova u standardizovane kodne strane 852, 855 i 866. Podržani su albanski, beloruski, bugarski, makedonski, mađarski, poljski, ruski, rumunski, srpski, slovački, slovenački, ukrajinski, hrvatski i češki. Ovo je posebno važno za nas budući da je pored latinice, (kodna strana 852) podržana i kompletna srpska ćirilica (kodna strana 855). Uz sve to paket sadrži i *Microsoft Chart* komplet ekranskih i printerskih fontova koji su prilagođeni podržanim kodnim stranama.



AlphabetPlus je projektovan kao dodatak Microsoft-ovim *International English* paketima, uz očiglednu želju da se na ovom novom i gladnom tržištu osnaži ulicaj čuvenog tekst procesora *MS Word* i ovaj eventualno učini standardom. Ovakve koncepcije se protežu i na ostale pakete – *Multiplan*, jedan od najjačih programa za unakrsno preračunavanja bi, uz *AlphabetPlus* mogao lako da omogući uniformno kreiranje tabela, finansijskih kalkulacija, faktura ili najbičnijih računa za muštenje u Pragu, Moskvi, Budimpešti i Beogradu. Gtavno „oružje" novog paketa leži u lakom prelapanju među jezicima – prilikom instalacije korisnik odlučuje koje aplikacije i koji jezik će da koristi kao osnovu, ali će kasnije u svakom momentu moći da pomoću samo jedne komande pređe u bilo koji drugi jezik. Na primer, sekretarica koja piše ugovori na srpskom će vilo lako moći da pređe na, recimo, mađarski ili poljski unese željeni tekst i ponovo se vrati na srpski.

Paket pored svega navedenog, sadrži drajvere za preko 70 različitih modela štampača uključujući Brother, Canon, Citizen, Epson, Hewlett-Packard, IBM, Star i još mnoge druge 9- 24-igline matrične i HP kompatibilne laserske štampače kao i latinične i ćirilične *download* fontove. Na 24-igličnim matričnim i laserskim štampačima korisnici mogu ćak da mešaju latinicu i ćirilicu na istoj strani i to *na svih 14 jezika*.

AlphabetPlus radi na svim PC-kompatibilnim računarima počev od XT-a sa najmanje 512 KB RAM-a, MS DOS-om 3.30 ili višim, EGA, VGA ili MCGA grafikom i hard diskom, a može se kupiti kod nacionalnih dilera po ceni od 170 DEM.

Kontakt adresa: Microsoft GmbH, Edisonstrasse 1, D-8044 Unterschleissheim, Deutchland, tel. 9949 89 3176 3831, fax 9949 89 3176-3800

Grafika

NOVOSTI IZ „HERCULESA"

"Hercules Computer Technology", ime koje je stvorilo prvi pravi grafički standard sveta personalaca, nastavlja svoju aktivnost. Posle izuzetno uspeha postignutog sa „Hercules Graphics Station Card" kao i nove i860-bazirane „SuperStation 3D" (detaljan test u narednom broju), „Hercules" je, u skladu sa trendovima, ubrzao dotle i svoju „Graphics Station Card". Nova „Hercules Graphics Station GOLD" takođe je građena oko 50 MHz TI 34010 sa 1 MB VRAM i VGA čipom i, uz visoke frekvencije osvežavanja (1024×768 na 70 Hz u 256 boja), omogućava i režime 800×600 u 32 768 boja i 512×480 (TARGA) u svih 16,7 miliona nijansi! Kod poslednjeg režima svaka tačka je 32-bitna – pored 24 bita za boju, sadrži i 8 bita za video preklapanje, transparentnost itd. Uz dodatni genlock, eto lepih video kreacija.

Kartica se po želji dobija i sa 2 MB display-list DRAM, s kojim će, uz priloženi ADI P386 drajver, u *AutoCAD 11* po brzini ekranskih operacija nadmašiti bilo koji 8514/A ili XGA adapter, jer se one u potpunosti odvijaju na procesoru i RAM kartice bez opterećivanja samog CPU ili sistemske magistrale. Očekujte prikaz u jednom od narednih brojeva.

Monitori

MONITORI NA DODIR

"GoldStar 1430Plus TouchMonitor" je puni naziv novog 14" VGA monitora osetljivog na dodir koji se, po želji kupca, prodaje serijskoj ili Bus verziji. Pored toga, izbor obuhvata i dve verzije površine osetljive na dodir: *IntelliTouch* koristi tehnologiju površinskih akustičkih talasa, a *AccuTouch* tehnologiju promene otpora. Prvi sistem se zasniva na transparentnom staklanom panelu kroz koji se propuštaju zvučni talasi koji bivaju apsorbovani dodirom prsta ili bilo kog mekog objekta, dok se drugi sistem zasniva na transparentnom poliesterskom sloju



Na dodir prsta: GoldStar 1430Plus Touch Monitor

iznad staklene površine, slojevi su razdvojeni sićušnim providnim tačkama (tanjim od 1/1000 inča), tako da najmanji dodir dovodi do promene otpora odnosno provodljivosti.

"GoldStar AccuTouch 1430Plus TouchMonitor" sa PC Bus kontrolerom košta $1160, a IntelliTouch verzija, takođe sa PC-Bus kontrolerom, $1280.

Kontakt adresa: Elographics Inc., 105 Randolph Rd., Oak Ridge, TN 37830, tel 615-482-4100, fax 615-482-4943.

Grafičke kartice

256 BOJA ZA WINDOWS 3.0

Firma *Aydin Controls* je konstruisala i pustila u prodaju novu grafičku karticu niske cene i velikih mogućnosti namenjenu korisnicima paketa *Windows 3.0*. Kartica nazvana jednostavno „7800", može da prikaže 256 boja bez prepitanja u rezolucijama do 1024 x 768, a uz to se može podesiti tako da obnavlja ekran frekvencijom od 70 Hz u cilju eliminisanja treperenja. Kartica je VESA i VGA kompatibilna i opremljena drajverima za *Windows 3.0*, *VESA* i *AutoCAD*, kao i posebnim utility programima za optimizovanje performansi, budući da koristi brzu logiku bez stanja čekanja. Sve rezolucije i formati prikaza se mogu birati softverski. Cena: 216 GBP.

Kontakt adresa: Aydin Controls Ltd., 1/3 Hunting Gate, Hitchin, Hertfordshire, SG4 0TJ, U.K., tel 44-462-458604, fax 44-462-456761.

Okrada teksta

"SIGNATURE" U PRODAJI

IBM i „XyQuest", firma poznata po tekstprocesoru *XyWrite III Plus* najzad su se sporazumeli oko distribucionih prava oko novog tekst procesora nazvanog *Signature*, koji su

Učenje na Sezamu

ELEKTRONSKA ŠKOLA

Pored svega što kao afirmisan konferencijsko-informstivni sistem već nudi, Sezam će svojim korisnicima uskoro pružiti mogućnost i da, ne izlazeći iz svoje radne sobe, diplomiraju u Americi

Jedne tople večeri, aprila 1985 godine, pisao sam seminarski rad iz predmete „Mediji i propaganda", u okviru postdiplomskih studija u Americi. Kao i sada, dok ovo pišem bio sam u škripcu s vremenom, pošto je rok za predaju rada isticao u ponoć. Red je govono o slici Jugoslavije u američkim medijima.

Bilo je već osam sati uveče i trabalo mi je pola sata da završim pisanje, a onda još desetak minuta da odštampam tekst. Verovatno bih stigao da završim rad odnesem ga na fakultet i predem sekretaru tekst negde do pola deset uveče, ali želja za avanturom kao i jeden poziv u bioskop bili su jači od mojih školskih obaveza.

Umesto štampanja jurnjave do škola i drugih dosadnih eporednih stvari ja sam uključio modem i u roku od nekoliko minuta ceo rad od nekih deselak strana poslao mom profesoru Paulu Levinsonu kao elektronsku poruku na jedan kompjuterski sistem na kome smo obojica bili pretplaćeni. Sistem je potvrdio prijem poruke i tačno vreme predaje, rok je bio ispoštovan, a moja žena i ja smo mogli u bioskop.

Sedam godina kasnije, po prvi put u Evropi, i vi to možete da uradite! Moguće je da zvuči neverovatno, ali od februara meseca ove godine u Beogradu postoji elektronski ogranak odseka za Teoriju medija američke škole *New School for Social Research* iz Njujorka. U saradnji sa nekoliko beogradskih firmi od kojih je jedna Sezam, oni koji su zainteresovani da dobiju američku međunarodno priznatu diplomu iz oblasti teorije medija mogu to od ove godine da učine i se ovih balkanskih prostora, ne napuštajući svoju radnu sobu i svoj računar.

Šta vam je potrebno da biste iskoristili ovu jedinstvenu priliku? Pored onoga što svi korisnici Sezama već pouzdano imaju – dakle računara (na poslu ili u kući) modema, komunikacionog softvera i telefonskog broja – potrebno je još nešto novca i odgovarajuća školska sprema. Sve ostalo je brige našeg (i američkog) organizatora. Zajedno se studenti i predavačime širom planete (u bukvalnom smislu) posredstvom centralnog kompjutera biće povezani u konferencijsku mrežu čiji se jedan ogranak nalazi na Sezamu u Beogradu. Jednostavno: vaš računar pozove Sezam, a komunikacija sa Amerikom je naša briga!

Vaši tekstovi, komentari i seminarski radovi (na engleskom) slažu se u konferenciju pod nazivom, na primer, „Kompjuteri i demokratija" koja funkcioniše na isti način kao i svaka učionica – predavač iznosi stavove o ovoj oblasti a studenti pitaju, komentarišu i odgovaraju. Na kraju semestra dobijate ocenu u skladu sa aktivnošću i kvalitetom učešća na „časovima". Razlika u odnosu na klasičnu učionicu je u tome što se profesor i studenti nikada ne susreću licem u lica, što se u normalnoj situaciji čak i ne čuju i što svako živi na svom kraju sveta i „dolazi" u školu kada to njemu najviše odgovara (zbog razlika u vremenu, profesori iz Njujorka bi morali da budu budni iza ponoći da bi radili sa studentima iz Beograda).

Varovatno se pitate kakve su prednosti ove vrste obrazovanje u odnosu na klasično? Pored onih koje su ovde već navedene, pomenuću glavne rezultate istraživanja sa Karnegi-Melon univerziteta, objavljene u časopisu „Scientific American" iz septembra 1991. Rezultati ukazuju da grupe koje komuniciraju elektronski za razliku od klasičnih a) dolaze do zaključaka sporije, b) iznose više ideja c) učestvuju podjednako u diskusijama i d) više se angažuju – sve u svemu, dobijaju kvalitetnije obrazovanje. Postoje i neke mnogo proziačnije prednosti: ako ste zaposleni ne morate da prekidate posao da biste studirali. Pored toga, štedite novac koji biste morali dati za put i život u inostranstvu – postaje vam pristupačno ono što je do sada izgledalo nedostižno! Na ovaj način dakle najkvalitetnije obrazovanje postaje dostupno i onim društvenim grupama koje to do sada nisu mogle da konste

Nastava koja se za sada održava samo na engleskom, već ne jesen će, u saradnji sa nekim od naših škola i univerziteta biti proširena i na predavanja na srpskom. Ako smo vas zainteresovali, javite nam se pošto ili preko Sezama, da bismo vam pružili dodatne informacije.

Što se tiče mog seminarskog rada sa početka teksta, mogu da vam kažem da je bio pročitan i ocenjen još pre nego što smo se mi vratili iz bioskopa! Rok i formalnosti su bile ispoštovane, mi smo videli film, a po povratku kući sačekala me je i obradovala „desetka" (A+ u Americi).

Dušan Tatomirović, M.A.

NAGRADE MAGAZINA BYTE

Tradicionalne BYTE-ove godišnje nagrade su dodeljene i pred nama su neki od dobitnika.

Da bi neki hardverski ili softverski proizvod uopšte konkurisao za jednu od nagrada, potrebno je da zadovolji određene uslove pre svega da je nastao, odnosno izašao na tržište tokom godine. Nominacije vrše urednici i saradnici BYTE-a imajući u vidu pre svega mogući uticaj novih proizvoda, inovacije, performanse i cenu. Prilikom glasanja, svaki urednik i saradnik dodeljuje određen broj glasova za 10 proizvoda. Proizvodi sa najviše glasova u svojoj kategoriji dobijaju prvu nagradu (Award of Excellence), proizvodi u sredini dobijaju drugu nagradu (Award of Distinction), a oni koji osvoje najmanje glasova dobijaju traću, počasnu nagradu (Award of Merit). Proizvodi koji ostanu bez jednog glasa, naravno, ne dobijaju nikakvu nagradu.

Svaki od 84 ovogodišnja pobednika ima određene dostignuća u svojoj kategoriji koja donosi industrijski napradak. „Apple" je, na primer, našao adekvatan odgovor na prošlogodišnjeg pobednika, *MS Windows 3.0*, u vidu svog novog operativnog sistema *System 7*. „Microsoft" je tokom prošle godine bio zauzet razvojem aplikacija i podrške za *Windows*, pa su se ove godine tri takva paketa, *Visual Basic*, *Word for Windows 2.0* i *Excel 3.0*, našli među nagrađenima. U portabl kategoriji su mnogi proizvodi dobili nagrade uključujući dva „Toshibina" notebook-a. „Psionov" ručni računar „Citizenov" najebook printer PN48 i „Zenith Mastersport 386 SL". Ipak, najvažniji i najznačajniji dobitnici nagrada u ovom domenu su proizvodi koji nisu namenjeni krajnjem korisniku – na primer „Chips & Technologies" i AMD-ovi procesori.

Appie System 7
(Award of Excellence)

Čekanje ne više puta najavljivan *System 7* se isplatilo. reč je o korisničkom interfejsu još veće lakoće korišćenja u odnosu na ranije izdanje. *System 7.0* sadrži poboljšane komunikacione funkcije, 32-bitni, QuickDraw, virtualnu memoriju, Ethernet i Token-ring drajvere, kao i mogućnost deljenja hard diskova i foldera sa ostalim umreženim korisnicima. Deo sistema je i *Interapplication Communication and Open Scripting Architecture*, koji obezbeđuje mehanizme za dinamičku deobu informacija i automatizovanje ponavljajućih poslova. Još jedna bravurozno delo „Appleovih" programera.

Visual Basic (Microsoft)
(Award of Excellence)

Sa *Visual Basic*-om programiranje u *Windows* okruženju postaje gotovo lako – *Visual Basic* kombinuje strukturalnu verziju bejzika sa korisničkim interfejsom za dizajniranje grafičkih alternate aplikacije. Dozvoljava kreiranje prozora, tekstualnih polja, komandnih dugmadi i ostalih grafičkih elemenata. Posle dvoklika na neki od elemenata (na primer meni) prelazi se u pisanje bejzik koda koji definiše funkciju svakog elementa. *Visual Basic* kombinuje bejzik kod sa kodom *Widnows* interfejsa. Za programiranje samostalnih *Windows* aplikacija (u paket je uključen i kompajler) više nije potrebno poznavati sve bubice *Windows*-a – dovoljno je poznavanje bejzika.

FoxPro 2.0 (Fox Software)
(Award of Excellence)

FoxPro je priznati brzinski šampion među *dBase* kompatibilnim programima za obradu podataka – nagradu je dobio kao najrazvijeniji program za verziju *2.0*. Nova tehnologija nazvana „Rushmore" prelražuje kroz baze sa milionima zapisa neverovatnom brzinom. Oslala poboljšanja uključuju kompromovane indekse, strukturalni jezik za pretrazivanje (Structural Query Language). WYSIWYG projekt menadžer, itd.

DR DOS 6.0 (Digital Research)
(Award of Excellence)

Potrebalo je izvesno vreme, ali „Digital Researchov" *DR DOS 6.0* je počeo ozbiljno da ugrožava MS-DOS, kao kralje leksi-baziranih operativnih sistema. DR DOS ima bolji rad sa memorijom, bolji *task-switcher*, i dokazane programe za keširanje diska i kompresiju fajlova. Pored svega toga, data je i doživotna podrška, a u paket je uključeno i nekoliko mrežnih alata, što bi moglo da bude posebno interesantno za nova *Novell* korisnike.

PC/Chip (Chips & Technologies)
(Award of Excellence)

PC/Chip predstavlja prvi pravi PC na jednom čipu i time je praktično sve rečeno. Po prvi put proizvođači računara mogu da naprave kompletan računar koristeći samo jedan čip i nešto memorije, što bi samo po sebi već bilo dovoljno za nagradu, ali C & T su dodali još jednu arhitekturnu inovaciju – SuperState načim da sa prošin PC arhitektura i dodaju nove mogućnosti bez uticaja na kompatibilnost ili pravljenje kompromisa u tom smislu.

Norton Desktop for Windwos (Symantec)
(Award of Distinction)

Kao esencijalni dodatak *Windows* okruženju. *Norton Desktop for Windows* potpuno zamenjuje trapavi *Program and File Manager* jednim mnogo elegantnijim desktopom. U paket je pored glavnog programa, uključena i cela kolekcija dodro poznatih utility programa.

Word for Windows 2.0 (Microsoft)
(Award of Distinction)

Još jednom se pokazalo da ljudi koji su napravili *Windows* najbolje razumeju koncept *Windows* aplikacija. Iako je *Word for Windows 2.0* prepun svih mogućih opcija koje se mogu zamisliti, najveći deo njegove slave ipak otpada na elegantni korisnički interfejs, koji omogućave maksimum posla uz minimum napora.

IBM: Optički drajv sa mogućnošću ponovnog pisanja
(Award of Distinction)

Na vrhu talasa novih tehnologija za smeštanje podataka su optički diskovi, ali da bi tu i ostali, potrebno je da prevaziđu jednu krupnu manu – jednom snimljeni diskovi se mogu samo čitati. IBM je uspeo da napravi kombinaciju magneto-optičkih i optičkih diskova od 3.5" koji omogućavaju ponovno pisanje i tako otvorio nove mogućnosti za distribuciju softvera i podataka, kao i za arhiviranje podataka. *(M K.)*

zajedno razvili, i program se krajem prošle godine našao u prodaji. Reč je o programu koji je zamišljen kao naslednik „XyQuestovog" *XyWrite III Plus*-a i IBM-ovog *DisplayWrite*-a. *Signature* je, zapravo, puna WYSIWYG verzija *XyWrite*-a obogaćena nizom novih funkcija i mogućnosti. Prvu novost predstavljaju meniji – *XyWrite*-ov zaštitni znak je bila komandna linija iz koje se moglo ostvariti sve što korisniku padne na pamet. Kombinacija komandne linije i pull down menija se pokazala kao najbolji kompromis između brzine komandne linije i preglednosti menija, pa je komandna linija zadržana i u *Signature*-u i čak dopunjena komandnim stekom.

WYSIWYG podrazumeva puni rad u grafičkom režimu, tonjove proizvoljno promenljive veličine (scaleable fonts), unošenje grafike (grafikoni, ilustracije, skenirane slike, itd) u tekst na proizvoljnom mestu i u proizvoljnoj veličini uz ulivanje teksta oko slika, delimisanje i postavljanje okvira, itd. Omogućeno je korišćenje PCX, TIFF i PCL formata a u paketu se dobija i pomoćni program za konverziju ostalih formata u TIFF. Jednom unetoj slici se može odrediti izrez, veličina i pozicija unutar strane, a iz *XyWrite*-a je preneta i mogućnost integrisanja grafičkih fajlova spremnih za štampu. Iako najspektakularniji, grafički način rada nije jedini – *Signature* omogućava i linijski, stranični, prošireni tekstualni način rada i rad sa slupcima, čime se omogućava brži i jednostavniji rad pri unošenju teksta.

Signature omogućava uvoz fajlova uz automatsku konverziju fajlova spremnih za štampu. Iako najspektakularniji grafički način rada nije jedini – *Signature* omogućava i linijski, stranični, prošireni tekstualni način rada i rad sa slupcima, čime se omogućava brži i jednostavniji rad pri unošenju teksta.

Signature je opremljen još nizom krupnijih i sitnijih opcija kao što su podrška miša, kondicionalni *Mail Merge* (vezano štampanje), povezivanje dokumenata, makro-procesor, crtanje linija, beležnica, generator sadržaja i indeksa itd. Jedna od stvari koje se svekako ne može nazvati sitnicom ali koja je potrebna samo određenim korisnicima je mogućnosi rada u mreži – podržane su *NET BIOS, Novell Advanced NetWare, IBM PC Nal, Banyan VINES* i *3Com 3+* mreže.

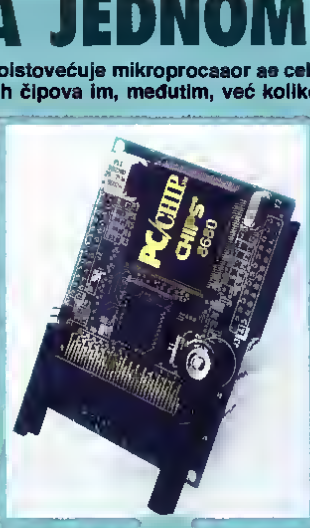
Signature radi na bilo kom PC i PS/2 kompatibilnom računaru sa najmanje 384 K RAM-a, hard diskom i DOS-om 3.3 ili višim i Hercules, CGA, EGA ili VGA video sistemom. Za instalaciju je neophodno obezbediti 3.11 MB slobodnog prostora na hard disku, zavisno od toga da li se instaliraju i outline fontovi, rečnici, helpovi i tutorijal, konverzioni filteri i ostali pomoćni fajlovi. Za komforan WYSIWYG rad se preporučuju minimalno 286 računar. Preporučene cene iznose $495 za nove korisnike ($295 za dodatne kopije) $119 za registrovane korisnike *DisplayWrite*-a i $99 za registrovane korisnike *XyWrite III Plus*. *(M K.)*

Kontakt adresa: XyQuest, 44 Manning Road, Billerca, MA 01821, U.S.A. tel. 508 671-0888

PC NA JEDNOM ČIPU

Većina ljudi, sasvim pogrešno, poistovećuje mikroprocaaor ae celim računarom. Najnovija kretanja u razvoju mikroprocesorakih čipova im, međutim, već koliko autra mogu dati za pravo.

Činjenica je da mikroprocesor obavlja većinu računskih operacija i određuje potencijalnu brzinu i snagu svakog personalnog računara. Mikroprocesor se toliko često pominje kao glavne odlika računara – *Upravo sam kupio 486 mašinu* – da se veoma često stvara predstava o mikroprocesoru kao „računaru na jednom čipu". Međutim, kompletan PC sistem zahteva veliki broj funkcija koja mikroprocesor nije u stanju da obavlja. Funkcije kao što su grafika rad sa diskom rad sa mamonjom – takođa određuju performanse sistema a za njihovo obavljanja su zadužani odgovarajući podsistemi. Većina ovakvih funkcija zahtava posebna kontrolera i ostale pomoćna čipove, i zbog toga se računar sastoji od velikog broja kompleksnih kola. Napredak na polju integracije čipova je od presudnog značaja za smanjenja „gužve" i smanjivanja dimenzija računara.



I ovo je XT: Pločica sa procesorom F8680

DODAJTE MEMORIJU...

U originalnim IBM PC računanma su sva pomoćna kola bila sastavljana od standardnih logičkih čipova zbog čega je matične ploča bila pretrpana. Nezavisni proizvođači PC kompatibilnih računara su doveli do razvoja specijalizovanih visokointegrisanih čipova (nazvanih *čip-setovi* po tome što zamanjuju ceo niz atandardnih čipova) čime su njihov broj, veličina i cene znatno smanjem. Današnji tipični PC kompatibilni računar se sastoji od mikroprocesora nekoliko čip-setova, nešto mamorijskih čipova i izvesnog broja ostalih komponanti.

Daljom integracijom, to jest kombinovanjem mikroprocesora i čip-seta na jednom čipu cena i dimenzije računara se dalja smanjuju uz istovremeni porast performansi, budući da takav čip objedinjuje sve osnovna funkcija sistema u jednom fabrički testiranom pakovanju. Ovako ostvarena prednost su navela niz proizvođača poluprovodnika na razvoj „računara na jednom čipu" (Single Chip PC). Prva ja na cilj stigla poznata firma Chips & Technologies" sa čipom čija je oznaka F8680 a njihov slogan glasi *Add memory and you have a PC* (dodajte mamoriju i imaćete PC). Ipak treba naglasiti da takav *Single Chip* PC raspolaže samo osnovnim funkcijama potrebnim za lormiranje XT ili AT kompatibilnog sistema, što znači da obuhvata mikroprocesor bazični video-kontroler, kontrolere memorije dijakova, tastature i I/O kontroler. Sve preko toge (na primer, pun VGA prikaz) zahteva posebna dodatna čipova. U sledeću generaciju *Single Chip PC*-a će najverovatnija biti dodeti i keš matematički koprocesor ubrzivači specifičnih funkcija kao što je grafike u **MS Windowa** poboljšani I/O kontroleri i kontroleri alternativnih uređaja kao što je SCSI. Za još dalju budućnost se predviđa integracija i takvih opcija kao što su podrška za Multimadia funkcije i kompresija pe ček i prepoznavanje glasa.

USPAVANI PROCESOR

Svaka od kompanija koje rade na ovakvoj integraciji ima sopstveni pristup problematici. Jedan od najzanimljivijih pristupa je „Intelov" – ova firma već ima veoma „kondenzovano" rešenje u vidu 386 SL procesora. Ovaj čip zajedno sa čipom 360 SL predstavlja bazu 386 SX računara sa jako redukovanom potrošnjom struje. 386 SL koristi modifikovanu formu 386 SX arhitektura koja omogućava inteligentnu kontrolu potrošnje. Jedan od načina na koji se to postiže je novi interapt nevidljiv za softver koji ima najviši prioritet u sistemu i koji omogućava procesoru da se prabaci u alternativni memorijski adresni prostor. Tu se izvršavaju kod i funkcija koja kontrolišu potrošnju, dok je ostali deo procesora potpuno zamrznut i nesvastan eksterne aktivnosti. Ovakav pristup omogućava potpunu kompatibilnost uz novu funkcionalnost a glavna mana je zahtev za redizajniranjem sistema u koje novi čip treba ugraditi.

"Intel" namarava da ide još dalja u integraciji i već najavljuje *Single Chip AT*, još uvek nije objavljeno kakva funkcije će imati takav čip ali je vrlo varovatno da će verno odslikati Intelovu težnju ka sistemima visoke klase. To najverovatnije znači keš VGA grafiku, a možda čak i malamatički koprocesor. Priča se da će *Single Chip PC* konceptu rade još i AMD i IIT. AMD koji je slekao pravo da prodaje sopstveni 386 procesor... svakako traži nova tržišta i verovatno radi na dizajnu 286 i 386 Single Chip sistema, dok će IIT kao poznati proizvođač matematičkih koprocesore najverovatnija napraviti čip sa ugrađenim koprocesorom.

NAJPRE XT...

Činjenica da je kompanija C&T prva uspela da proizvede takav čip ne iznenađuje kada se ima u vidu njihovo veliko iskustvo u proizvodnji čip-setova. Pravo iznenađenje predstavljaju primenjena arhitekturalne inovacija. F8680 PC/Chip se sastoji od jednog 8086 mikroprocesora 8524-kompatibilnog tajmera 8259-kompatibilnog interapt kontrolera, mehanizme za emulaciju DMA kontrolera XT tastatura CGA grafika, mamorijskog kontrolera, 16C450-kompatibilnog univerzalnog asinhronog risivera/transmitera za serijske komunikacije i standardnog 8-bitnog XT basa.

Poređ toge, čip sadrži i novi operacioni mod nazvan *SuperState R*, koji je sličan alternativnom modu u 386 SL, ali je i mnogo fleksibiniji. F8680 automatski ulazi u ovaj mod prilkom uključenja ili reseta, a omogućan je i pristup iz svakog hardverskog ili softverskog interapta, pomoću I/O instrukcija, tajmera ili bilo kog od četiri eksterna programirana signala. Razulilat je fleksibilna arhitektura koje omogućava obrazovanje virtuelnih uređaja upravljanje periferijama i emuliranja drugih procesora.

Deo za kontrolu grafike podržave klasične monitore i LCD ekrana, potpuno je CGA kompatibilan uz monohromatsku rezoluciju od 640 x 200 ili 640 x 400 ili 320 x 200 ili 320 x 400 u četiri boje. Poređ toga, na raspolaganju je i *Visual Map* tehnologija koja omogućava preciznu mapiranje boja i prikazivanje 16 nijansi sivog na LCD ekranu. Kontroli potrošnje energije je posvećena velika pažnja u dizajnu F8680 koji konsti samo statička kola. Mikroprocesorsko jezgro može da radi na bilo kom naponu u rasponu od 2.7 do 5 V, dok ostale komponente rada samo na 5 V. Minijeturizacija poluprovodničkih uređaja ave jasnija naginja ka novom standardu od 3.3 V i kako ovakvi delovi budu postajali dostupni, tako će i dizajn celog čipe slediti ovaj trend. Stvar je jasna – čip koji radi na 3.3 V troši približno 40% energije u odnosu na 5-V čip. Statički dizajn F8680 omogućava potpuno zaustavljanje sistemskog kloka a moguće je i daljenje taki-signale. Isko de sistem silazi u niža stanja i troši manja energije. Jedne od takođe bitnih prednosti F8680 u odnosu na standardni 8086 ili čak 286 je činjanica da koristi 26 bitna memorijske adrese što mu omogućava pristup do čak 64 MB RAM-a. Pri tome koristi preklapenja između banaka i sposoban je da se prilagodi svim vrstama memorije od potpuno statičke do potpuno dinamičke uključujući i PCMCIA memorijske kartice. F8680 može da radi na taktu od 0 do 14 MHz uz limitiranje maksimalnog takta pri niskonaponskom radu.

...A ZATIM I AT

Sledeći logičan korak je očigledno projektovanje *Single Chip 386* sistema što će najverovatnija pre stići iz „Intela" ili AMD-a nego iz C&T-a, ali i C&T radi na takvom čipu. Reč je o naprednom 386 klonu koji uključuja varziju SuperStata mehanizma nazvenu *SuperStale V*. Tipični današnji desktop sistem se sastoji od 386-bazirana mašine se VGA grafikom i najčešće kešom. Sve to uz standardni čip-set koji se konsti u takvom sistemu treba integrisati na jedan čip, možda čak i zajedno sa logikom potrebnom za EISA ili Micro Channel arhitekturu. Mnogo zanimljivyi su budući prevci razvoja koji bi vodili do inkorporacije novih arhitektura i modova kao što je SuperState. Jedna od mogućnosti je uključivanja mrežnog interfejsa čime bi se izbegli mnogi problemi koji danas najčešće iskrsavaju prilikom priključivanja računara na mrežu. U svakom slučaju jedino što je sigurno u vezi *Single Chip PC*-a je da će na tržištu proizvesti napredivive slekte.

Pripremio Marko Kirić

NAJMANJI NA SVETU

U svetu parsonalaca prisutna su dva glavna trenda razvoja: trend performansi i trend minijaturizacije. U prvom slučaju, put napretka je tekao od 8088, preko 8086, 286, 386, do 486, a uskoro i 586 procesora, kao i sve kapacitetnijih memorija, bržih diskova i kvalitetnija grafika. U drugom, posle standardnih desktopa došli su portabl računari, laptop, notebook, palmtop, a sada i mnogi „hand-held" (ručni) personalci. Prvi, uslovno rečeno, pravi DOS PC u ovoj klasi je HP 95LX.

Nebojša Novaković

Na prvi pogled, HP 95LX veoma podseća na mnogobrojne japanske džepne računare namenjene poslovnim ljudima sa funkcijama elektronskog rokovnika. Sve sličnosti se tu i završavaju. „Hewlett-Packardov" mališa dimenzija 16 x 8,6 x 2,5 cm i mase 312 grama, po snazi i podršci višestruko nadmašuje japanske igračke. Mada to ne znači da neke mane ipak nisu prisutne.

"Hewlett-Packard" kod svih svojih proizvoda drži konstantno visok kvalitet dokumentacija. To važi kako za Vectre 486 i HP Apollo 700 grafičke radne stanice, tako i za 95LX. Dokumentacija za 95LX sastoji se iz dva knjige. Prva je „Quick Start Guide" na oko 50 stranica, priručnik za prvo upoznavanje mogućnosti računara, a kada to prođemo, veliki „User's Guide" na oko 650 strana trebalo bi da nam otkrije svaki detalj i mogućnost 95LX.

HP 95LX je građen oko već pomalo zastarelog NEC V20H CPU, koji je nekada bio prilično popularan među vlasnicima XT-a. Ovaj procesor ima nešto ubrzano izvođenje naredbi u odnosu na obični 8088, pri čemu su posebno ubrzane naredbe za celobrojnu aritmetiku. To je, pored izuzetno male potrošnje ovog CMOS čipa, mogło biti odlučujuće za HP prilikom izbora procesora. Takt procesora je 5,37 MHz, što, uz memoriju sa 0 stanja čekanja, daje brzinu rada oko 30-40% veću od originalnog 4,77 MHz 1 ws IBM XT.

Za današnje prilike, kada već imamo notebook računare sa 33 MHz 486, ovo može biti smešno, ali će za namene za koje se ručni PC koristi ova brzina biti dovoljna.

Ono što kod specifikacije 95LX na prvi pogled pada u oči je ceo megabajt ugrađenog ROM-a. U ROM-u je, pored MS-DOS 3.22, i *Lotus 1-2-3* verzija 2.2 i niz drugih korisnih aplikacija, tako da je 512 Kb ugrađenog RAM-a najvećim delom slobodno za podatke ili konščenje kao RAM disk. Sedržaj tog RAM diska se čuva i po isključenju računara, tako da i po oznaci (C:) i po brzini podseća na meki hard disk. Standa veličina RAM diska je 254 Kb, s tim što se raspodela memorije između logičkog RAM-a i RAM diska konsnički može vršiti iz SETUP programa. Ako vam je potrebno više prostora, ugraden JEIDA slot prima RAM kartice kapaciteta 128 ili 512 Kb, s tim što se ona koristi kao RAM disk A.

Najslabija tačka HP 95LX je svakako ekran. Mali LCD dijagonale 12 cm prikazuje 40 kolona u 16 redova, pri čemu je matrica slova ona koja je uobičajena kod džepnih računara 5 x 7.

Grafička rezolucija je 240×128, što je dovoljno za neke veoma proste grafove u 1-2-3 ili elektronski rokovnik, ali ništa više. Naravno, ekran nema pozadinsko osvetljenje niti će se tome moći tako skoro nadati u klasi ručnih personalaca. Opet, na ovako malom prostoru veći ekran u ovom trenutku i nije moguće smestiti.

Tastatura predstavlja „Hewlett-Packardov" pokušaj da očuva kompaktnost tastature sa svojih džepnih računara, a da je pri tom učini prihvatljivom i za korisnika PC-ja. Tastatura je zaista izuzetno kompaktna, ali prihvatljivost za ljude navikle na PC tastature je pod velikim znakom pitanja, kao i za sve računare ovih dimenzija.

Od ugrađenih komunikacija HP 95LX poseduje RS 232 serijski interfejs sa modifikovanim konektorom, kao i minijaturni infracrveni predajnik za bežičnu 2400 bps komunikaciju dva 95LX na razdaljini do 20 cm, kao i za HP štampač za džepne računare.

Pored MS-DOS, HP 95LX u svom velikom ROM sadrži i *Lotus 1-2-3* V 2.2, donekle modifikovan zbog manjeg ekrana. Sa programom, koji ne ima svoj poseban taster za startovanja na tastaturi, rukuje se relativno lako. Ostale ugrađene aplikacije su rokovnik, telefonski imenik komunikacije sa PC, HP kalkulator, memo editor i prosti FILE MANAGER nalik na onaj u slanim verzijama PC *Tools*-a. Ako doplatite *PC Connectivity Pack*, pored serijske veze sa PC dobijate i PC verzije ovih paketa, tako da se podaci direktno prenose u vaš stoni računar.

Najmanji PC na svetu, što HP 95LX u ovom trenutku svakako jeste, rezultat je niza pažljivo izbalansiranih kompromisa u dizajnu koji su omogućili da u unutrašnji džep vašeg sakoa stane stvar koje se sigurno po snazi ne može poreditи ni sa notebook računarima, a kamoli sa stonim grdosijama, ali koja radi svoj posao. A njen posao je da bude pouzdan elektronski pomoćnik užurbanom poslovnom čoveku koji nema vremena za gubljenje. Kad u obzir uzmemo i cenu koje stoji iza ovog ne baš jeftinog proizvoda, jasno je da povereње može biti dato.



Ima sve što treba: U ROM računara ugrađen je *Lotus 1-2-3*, rokovnik, telefonski imenik, memo editor, HP finansijski kalkulator, komunikacioni softver (uključujući i XMODEM protokol), modem i interfejs za povezivanje sa PC računarima kupuju se posebno

Korisna adresa

MICRO AIR
11000 Beograd, Knaza Miloša 17
Tel/fax 453 352

U SVIM BOJAMA DUGE

"Hewlett-Packardov" novi printer nudi grafiku u boji i rezoluciju od 300 puta 300 tačkica za 700 dolara. Jedan kratak susret je pokazao koliko specifikacije na papiru ispunjavaju očekivanja.

Vladimir Kostić

"HP Desk Jet 500C" je veoma lepa igračka. Štampa tekst i grafikone u boji u rezoluciji laserskog printera. Posle par odštampanih ekrana iz *Windows*-a, svako bi poželeo da ima „HP 500C" na svom stolu.

HP odavno proizvodi takozvane *ink jet* printere. Glava se sastoji od minijaturne rešetke iz koje izleću kapljice mastila. *Ink jet* printere karakteriše praktično bešuman rad, zadovoljavajuća rezolucija i pristojna brzina. Ranији modeli su najbolje rezultate postizavali se specijalnom hartijom, koja dobro upija mastilo. Tehnologija je, naravno, napredovala, pa „HP 500C" zadovoljavajuće štampa na bilo kojoj hartiji čak i na providnim folijama. Ono što razlikuje „HP 500C" od ostalih *ink jet* printera (uključujući i crno belog brata „HP 500") je sposobnost da štampa u boji. Jeftino. Odmah da kažem da „HP 500C" lepo štampa u boji. Odštampani grafikoni sigurno ne liče na brljotine.

LEP IZGLED

"HP 500C" spolja izgleda potpuno isto kao model 500, o kome su „Računari" pisali u prošlom broju. Na stolu zauzima pristojno mesta. Dizajn je vrlo „oštkast" – svi uglovi su 90 stepeni. Zaobljenih površina nema nigde.

U radu, „HP 500C" se ponaša kao klasičan laser. Printar napunite hartijom (oko stotinak listova A4). Magacin sa hartijom ne zalcet, nije mnogo veliki. Ako mislite da štampate program od 20 000 redova, moraćete da izigravate *baby sitter*-a.

Pozadi se nalaze Centronics i RS232 portovi. Koristite onaj koji želite. Rad printera je skoro potpuno bešuman.

"HP 500C" sadrži tri fonta – Courier, Times i Gothic – u veličinama od 6 do 12 tačaka, a mogu se koristiti i bold i italic efekti. Naravno, pošto „HP 500C" podržava grafiku u rezoluciji 300 puta 300, nema problema da iz Venture štampate kakva hoćete slova.

Kvalitet otiska je problematičan. Razolucija od 300 puta 300 asocira na lasere, ali otisak više liči na 24-pinski matrični printer. Slova su nekako debela, neprecizna i nemaju onu oštrinu koju ima laser. Isto važi i za grafiku, bilo da je crno- bela ili u boji – preciznija je od onoga što proizvode matrični printeri, ali nema onu kristalnu oštrinu koju ima laser. Ne kažem da je otisak loš, samo nije kao kod lasera – avidalo se to „Hewlett-Packardu" ili ne.

TAJNA BOJE

Ispod poklopca, „HP 500C" krije glavu koja sa šest levo-desno kao kod matričnih printera. Glava sadrži kartridž se 3 mastila: ljubičasto, svetloplavo i žuto. Kombinacijom te tri boje, odnosno sa više prolaza glave, može se napraviti gotovo neograničeno puno boja. Ako se poklopac drži otvoren, tačno može da se vidi kako nastaju nijanse. Stvar lepo radi – ako vam ne treba super kvalitet raprodukcije boja.

Naravno, i crna boja se postiže sa više prolaza glava. „Hewlett- Packard" je razumno zaključio da će „HP 500C" često štampati samo crno beli tekst, a to bi trajalo i trajalo. Zato se color kartridž može zameniti sa crnim. Sa crnim kartridžom printer postiže brzinu od oko 2,5 stranice u minuti teorijski naravno. Nije baš brzo. Štampanje u boji, naravno traje mnogo duže. Zavisno od kompleksnosti grafika, zasićenja boja i kvalitete koji se žeti postići, može da traja od 30 sekundi po stranici pa do preko 90. Ponekad i duže.

"Hewlett-Packard" isporučuja drajvere za *Lotus 1-2-3, Harward Graphics, Quatro Pro* i još nekoliko popularnih aplikacija. Naravno, drajvar za *Windows 3.0* nije zaboravljen. *Windows* drajver omogućaje da se vrlo fino podesi štampanje boja i grafike, odnosno da se korisnik opredeli između brzina i kvaliteta. Posle nekoliko proba došao sam do zaključka da je žaljenje žrtvovati kvalitet. Treba se, dakle, naoružati strpljenjem.

Jezik kojim se „HP 500C" kontroliše nije *PostScript II* na žalost. „Hewlett-Packard" se opredelio za sopstveni jezik iz istog razloga iz kog ne koristi *PostScript* u svojim laserskim printerima – da uštedi na ceni. Upotreba *PostScript* jezika garantuje široku kompatibilnost i velika mogućnosti ali i zahteva skupu licencu koju treba platiti firmi „Adobe". Takođe. *PostScript* zahteva jak procesor ugrađen u printer (često RISC) i puuuuno memorije.

Srećom, većina aplikacija koje rade sa bojom će sigurno podržati „HP 500C". Ako aplikacija radi pod *Windows*-om *3.0* onda je „HP 500C" već podržan sasvim dobro. Najzad stari „HP 500" podržava veliki broj aplikacija, što garantuje da će „HP 500C" moći da se koristi kao crno beli printer, ako ništa drugo.

Vlasnicima „mekintoša" je namenjen model „DeskWriter C", potpuno istih karakteristika, samo softverski prilagođen „mekintošu".

Cena je, zvanično, 1095 dolara, za „HP 500C" i „HP DeskWriter C" Međutim, po radnjama se cena kreće nešto iznad 700 dolara. Za kolor printer opisanih karakteristika, nije skupo. Ono što jeste skupo je kartridž, bilo crni ili u boji. Cena je 20 dolara, a crni može da odštampa samo 500 strana, onaj u boji još manje.

Lepa stvar je da se kartridž menja za nekoliko sekundi, a da se pri tom ne zamažete mastilom kao prase.

PRLJAVE IGRE

Osim za štampanje grafikona i dampove ekrana, „HP 500C" se teško može upotrebiti za nešto pametno. Osim Uzmete skener u boji (ručni modeli su postali neverovatno jeftini). Skenirate novčanicu od 20 dolara. Odštampate je na „HP 500C". Odete do najbliže stanice metroa. Ubacite tih lažnih 20 dolara u mašinu za automatsku prodaju karata. Izabereta kartu od 16 dolara (ne može manje). Pokupite 4 dolara kusur (u pravom novcu!) Ponovite operaciju jedno desat puta. Probao sam – što radi i kažnjiv je zatvorom.

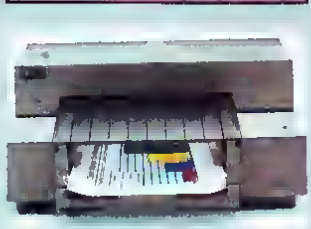
Koliko je „HP 500C" dobar zavisi od namene. Za kvalitetan crno-beli tekst niti je dovoljno kvalitetan niti brz. Za grafikona u boji i dampova ekrana zadovoljava, uglavnom. Za stvarno vrlo kvalitetnu grafiku u boji, treba potražiti neki od renomiranih „Tektronixovih" printera, ali tu cena počinju tek od 10 000 dolara. Najzad, za tehničke crteže: ploter sigurno neće doživeti konkurenciju u vidu „HP 500C".

Ostaja preslikavanja ekrana iz *Windows*-a.

Igračka. Skupa, ali lepa.

Ljične karte

Model HP 500C
Proizvođač Hewlett Packard Co. 19310 Pruneridge Avenue, Cupertino California 95014, USA. Tel 1-800-752-0900
Cena Zvanično 1095 dolara, ali se može naći za nešto iznad 700
Opis *Ink jet* printer sposoban da štampa u neograničeno puno boja. Namenjen IBM PC kompatibilnim računarima
Kvalitet štampe Otprilike kao kod 24-pinskih printera
Brzina Oko 2,5 stranice u minuti crno-belo, u boji zavisi od kvaliteta koji se želi postići i može da traja u prosaku od 30 do preko 90 sekundi po stranici
Razolucija 300 puta 300 tačaka po inču
Drajveri HP isporučuja drajvere za *Windows 3.0, Lotus 1-2-3, Hardward Graphics* i još nekoliko popularnih programa
Programski jezik HP (nije *PostScript II* kompatibilan)



Za i protiv

Za
- Lepo radi i na koška puno
- Tradicionalni HP kvalitet izrade
- Dušu dalo za dump ekrana i grafikona u boji

Protiv
- Crno beli takst ne može da se poradi se laserom
- Mastilo je skupo

UJEDINJENE BOJE PAKARDA

Prs nego što se pojavio najnoviji Hewlett-Packard skener, model ScenJet IIc, bili smo skloni da mislimo da je kolor fotografija teško dostižan cilj u stonom izdavaštvu. Nakon iakustava koje smo doživali sa ovim uredajem i opsežnih testova koje smo obavili, verujemo da smo svedoci još jedne od malih revolucija kojom se brišu razlike izmedu mogućnosti industrijskih mašina i svakom dostupne personalne računarake tehnologije. Pojave ovog skenera predstavlja značajan datum i za ljude koji se beve računarskim prezentacijama i dokumentarističkim poslovima.

Zoran Životić & Dejan V. Veselinović

Imati skener do skoro je, čak i za one koji se bave stonim izdavaštvom, bio prilično luksuz, ne toliko zbog seme cene koliko zbog odnosa cena/konkretne primenjivosl. Skener je bio nezamenjiv kod poslova koji su zahtevali uključivanje jednoslavnijih crteža (pre svege zaštitnih znakova firmi dijagrame i slično) i u poslovima više okrenutim samom dizajnu – prosečan posao slegenje je teško mogao da isplati investiciju u skener. Pojavom dovoljno jeftinih i dobrih OCR programa, skenerima je otvorene nova dimenzija primene na polju unosa teksle. Konačni udarac svekoj dilemi da li će skeneri doživeti masovnyu ili ber češću upotrebu zadaje pojave kolor skenera.

UKLJUČI I POČNI

Prvo što čoveka impresionira u slučaju Hewlett-Packard skenera sa oznakom „ScanJet IIc" je njegova veličina da baš ne kažemo gigmaznosl. Iza površine skeniranja od 216×356 mm (ŠxD) stoji prilično niska kutija, ali je dublje od bilo čega se čime smo se do sada suareli. Skener nije ni lagen, ima vrlo pristojnih 13,4 kg.

Sve komende za rukovanje skenerom smeštene su ne dve diskele – na samom skeneru nalaze se samo brava za imodilizaciju glave lokom transporta, utičnice za veznike i taster za uključivanje i isključivanje. Sve ostalo se radi čisto programski. Elegeniino, nema šle.

Pre nego što upotrebite skener, morate ubaciti u svoj računar veznu karticu. Ona je ä-bitna, pa vas neće oštetiti za neku dragocenu 16-bitnu utičnicu. Preme dokumentaciji HP-a, to je jedna od varijanti SCSI veze, ali HP napominje de ne garantuje da će kartica raditi sa bilo kojim drugim SCSI uređajem. Kartica je lepo uređena i praktično se sastoji od jednog jedinog VLSI kola sa oznakom NCR i jedne grupe od 4 klip-klep preklopnika, koji služe za odredivanje adrese na kojoj će se skener naći. Da se ne bisle previše mučili jedan od programa koji HP prilaže uz skener će ispitati sve i obavestiti ves keko treba da podesite karticu. Kažu da u većini slučajeva sasvim dobro obavi posao.

Ako koristite neki od programa za upravljanje memorijom (e da nije sada već mellene obavezni EMM386.SYS), možete se susresti sa (ne)očekivanim leškoćama. Međutim (ineče odične) dokumentacija se bavi isključivo problemima i rešenjima za *Windows* (jer ge konsli kao podlogu), a o ostalime ni reči.

U prvom pokušaju suderili smo se sa (očekivanim) problemom, jer je naš *386-to-the Max* popunio sve rupe i rupice u memoriji, pa skener nije imao gde da se smesti. Problem smo rešili leko što smo uključili obavezni SYS red u CONFIG.SYS, e zatim provrteli krug sa polje programom MAXIMIZE, koji je jednoslavno



Nenad Marijanović

Hewlett-Packard ScenJet IIC

ukljućio i ovu poduđu u one koje treba prebaciti iznad granice od 640 KB – i sve je lepo proradilo.

Programska instalacije je jednako bezbolne naprosto ubacite disketu, udete u *Windows 3.0* i odalle ga instalirate. Sve je potpuno automatizovano i izgleda bez greške – instalirali smo program na tri računare u različitim konfiguracijama i sve je radilo kao pesma. Kao što i očekujemo od HP-a.

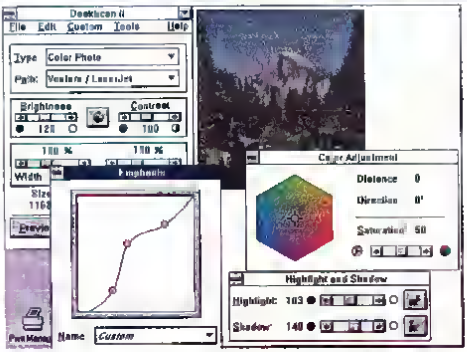
DESKSCAN II

Programska podrške koja stiže uz ovaj skener je daleko snažnija od onoga što smo do sada imali prilike da vidimo uz uvekve uređaje. Naravno, kao što se moglo očekivati na osnovu onoga što je bilo deto još uz prvi HP skener okruženje je opet Microsoft *Windows*. Paket se sastoji iz dve bitne dela – prvi je *DeskScan II* program nemenjen samom skeniranju, a drugi je *Publisher's Paintbrush*, za delju manipulaciju sa slikama.

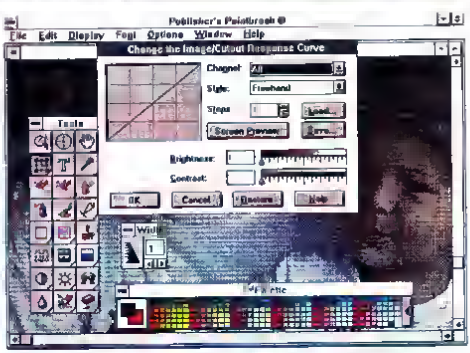
DescScan II je poboljšana varijanta standardnog softvera koji je HP isporučivao uz ranije modele. Reći da je samo pobolišana, čak i kade se gledaju samo one opcije koje su postojale i ranije, nije najpravednije prema ovom softveru, jer izgleda da se ipek radi o potpuno novom programu, koji sa samo po osnovnoj ideji oslenja na ranju koncepciju. Vlasnici *ScanJet*-a znaju da je ovaj program, iako korektno obavlja posao daleko od funkcionalnog savršenstva i da je praktično neupotrebljiv kod malo veće količine posla. Probnog skeniranje je davalo malu sliku, na osnovu koje je bilo jako teško poslaviti uzorak pod pravim ugiom ili precizno iseći manju površinu. Rad je bio prilično spor, a recimo izbor razmere tako da konačna slika zauzima unapred zadatu površinu je predstavljao pravu noćnu moru. U novoj verziji *DeskScan*-a sve ovo je potpuno drugačije.

Ako za trenutak zanemarite nekoliko dodataka koji se odnose na boju i koji se kriju pod meni opcijom *Tools*, odmah ćete uočiti niz drugih novosti. Prve koja će vas svakako odradovati je mogućnosl zumiranje već kod probnog skeniranja – uokvirite deo površine koji vas zanima i taj deo će biti prikazan preko celog radnog dela ekrane. Ukoliko vam je i ovo nedovoljno za postavljenje uzorka pod pravim ugiom (da honzontalna linija bude zaista horizontalna) i precizno isecanje i samu površinu za probno skeniranje možete povećati sve do prostora koji ostaje kada se od ukupnih dimenzije ekrana oduzme površina se kontrolema. Pored od ranijeg poznatog izbora razmere preko procenta povećanja/smenjenja, sada imamo i opciju kojom se direkino zadaje dimenzija slike koju želimo da dobijemo od izabranog isečka. Jedinica mere se naravno mogu birati izmedu milimetara, inča, lačaka i šlamparske jedinice punkta. Odmah kod probnog pregleda možete menjati kontresl i globalno osvetljenje slike, izvršiti okretanje slike kao u ogledalu i inverziju u negativ. Ukoliko ste aktivirali opciju **Live preview**, odmah ćete ne ekranu primetili kekve efekte ove promene izazivaju.

Naravno, ključne izmene u programu su vezane za samo skeniranje u boji. Već na osnovnoj kontrolnoj tabli možete birati tip bi nime slike koju želite da dobijete prema uzor-



Komandna tabla programa *DeskScan II* sa aktivnim alatima. Podešavanje boja, naglašavanje, osvetljena mesta i senke (u radnom prostoru se nalazi demo fotografija koja se dobija uz skener)



Komandna tabla programa *Publisher's Paintbrush*. Bogata garnitura alatki omogućuje neograničeno manipulisanje kolor slikom (u radnom prostoru se nalazi fotografija u rezolucijii 100 dpi)

ku koji skeniraše. Opseg se kreće od crno belog ili kolor crteža preko crno belog ili kolor rastera i crno bele ili kolor fotografije do specijalitete ovog skenera koji je zapravo i najnovijina novina a HP ga nazva *Millions of Colors*

Ova opcija se oslanja na ono što se polako nameće kao standerd u računarskom treliranju boja i već ranije je predstavljeno na mekintošu – boje se predstavlja sa tri bajta od kojih svaki pamti 256 nijansi svoje boje. Dakle svaka skenirana tačka je predstavljena sa 24-bita informacije (nešto više od 16 miliona kombinacija), što je za trenutne standarde u raspoloživoj memoriji na personalnim računarima, prilično kritično i očigledno predstavlja korak ka budućnosti. Prevedno na neke konkretnije mere ova količina informacija znači da jedna A4 strana skenirana u punoj rezoluciji od 400 x 400 tačaka po inču i sa 24-bitnom kolor informacijom traži datoteku od oko 42 MB! I sama tipična kolor fotografija 9×12 cm zahteva prostor od oko 8 MB. Naravno sa ovakvim količinama nije lako raditi na PC računaru a najčešće nema ni potrebe. S obzirom da nema smisla skeniranje vršiti vecom rezolucijom nego što je izlazni uređaj u stanju da prikaže, a posebno zbog toga što se pilenje rezolucija kod kolor fotografija nešto drugačije tretira. *DeskScan* nudi opciju kojom se skeniranje može prilagoditi izlaznom uređaju i cela kombinacija zapamtiti pod proizvoljnim imenom kao buduća „staza za štampanje" (*Print path*). Ova opcija u sebi krije na samo prilagodavanje rezolucija, već i jednu izuzetno važnu stvar – kalibraciju skenera prema izlaznom uređaju. Program prvo kreira datoteku (ili direktno štampa) test uzorak i to crno beli i opciono kolor koji se, nakon što ih odštampala na konkretnom izlaznom uređaju poslavljaju na skener i skeniraju. Znajući tačno kakav uzorak je kreirao i videvši šta je posle procese štamparije od toga ispalo, softver ja u stanju da izvrši kalibraciju kako bi skener u buduce davao optimalne rezultate

Koliko je ovo pitanje važno shvatićete kada pokušate da ručno uličete na konačan izgled. Na raspolaganju su vam tri opcije iz menija **Tools** (Alati) Opcijom *senke* i *osvetljena mesta* (**Highlight and shadows**) mozete korigovati ekspoziciju slike i to na dva načina – promenom numeričkih vrednosti ili, daleko interesantnije ukazivanjem na najtamniju i najsvetliju tačku na tolografiji nakon čega se čitava ekspozicija koriguje tako da ova dva ekstrema padnu u zona u kojima se još uvek naziru detalji. Oni koji se bave fotografijom i upoznali su sa zonskim sistemom podele logaritamske krive zacmjenja prepoznace da se ovde radi o korekciji koja senke treba da dovede u zonu II a najsvetlije detalje u zonu VIII. Sve međuvrednosti se potom koriguju između ova dva ekstrema

Ukoliko ovo podešavanje nije dovoljno i imate potrebu za preciznijim intervencijama na svakom delu knve zacmjenja na raspolaganju je opcija *naglašavanje* (**Emphasis**) Na ekranu dobijaje pravu sa tri kontrolne tačke na ravnomernim udaljenostima. „Hvatanjem" tačke i pomeranjem u bilo kom pravcu deformišele liniju u novu krivu zacmjenja koja će biti korišcena za korakciju originala

Svakako najetraktivnija opcija je podešavanje boje Na ekranu se pojavljuje šestougaona površina u kojoj svaka stranica predstavlja jednu od sledećih boja žuta, crvena magenta, plava, cijan i zelena – između su pralazi od jedna do druge boje. Pomeranjem kontrolne tačke iz centra prama nekoj od stranica vršimo korekciju naglašavanjem boje ka kojoj se krećemo dok se intenzijai promana definiše udaljenjem od centra

Kada sve vrednosti podesita, možeta konačno zadati finalno skeniranje. Brzinu skeniranja nismo precizno merili, jer se pokazalo da ona u najvećem broju slučajeva uopšte i ne zavisi od skenera. Brzina će naravno u najvećoj mari zavisiti od parametara koje smo izabrali jer je jasno da za onih pomenutih 42MB treba prilično vremena da se „proguraju" kroz interfejs, pa i da se zapišu na disk. Glave skenera se najčešće kreće u koracima sa pauzom jer je propusna moć samog skenera veća nego cele kombinacija interfejs/program/disk koja iza njega sledi. Ipak u nekim svakodnevnijim situacijama kao što je skeniranje fotografije 9×12 sa 24-bitnom kolor informacijom u razolucijr 100×100 dpi, nećete potrošiti više od 5 do 10 sekundi na finalno skeniranje

Izbor formata u kome možete snimiti sliku nije prebogat ali je sasvim usaglašen sa ostalkom softvera za *Windows* okruženje – na raspolaganju su vam PCX TIF BMP i EPS. S obzirom na svežinu standarda i prilagođenosi 24-bitnoj kolor informaciji najčešći izbor kod snimanja će biti TIF format. Slike u ovom formatu se leko razmenjuju sa *Publisher's Paintbrush*-om koji sa Jakods isporučuje uz skener i sa *CorelDraw*-om koji je drugi logičen izbor za dalja korišćenje slika.

PUBLISUER'S PAINTBRUSH

Dobro poznata firma *ZSoft*, koja sa prosla vila svojim *Paintbrush* programom i uvela standardni PCX format sulor je drugog programa iz skener pakata. Program je, naravno sasvim standardan za manipulaciju crtežima na računaru ali ono što ga čini posebnim je upravo podrška 24-bitnom kolor formatu koji daje skener. Koliko je ova činjenica ulicala na samu koncepciju najbolje ćete uočiti kada izaberete opciju delimisanu sličicom pipete. Poslavila pipetu na neku tačku slike i kliknita mišem – program će u kolor palati poslaviti kao izbor onu boju koja se krije iza tačke – metodom procena ovo je praktično nemoguće. Količina boje kojom program manipuliše čini skoro potpuno izlišnom bilo kakva lokalne ručne manipulacije slikom na koje smo nevikli u crtačkim programima. *Publishers Paintbrush* je zato daleko više orijentisan na manipulaciju slikom kao celinom nudeći da i tako oslejive opcije kao što je uklanjanje sitnih mrlja sa slika ukloni sam, analizom površine

Bilo bi najpravednije prikazati ovaj program u jednom zasebnom tekstu, jer se njime mogu kreirati vaoma atraktivne kolor ilustracije polpuno „od početka". Ipak lakav prikaz ćemo ostaviti za neku drugu priliku koncentrišući se u ovom tranutku na obradu skeniranih kolor fotografija

Program je urađen u pravom *Windows* 3.0 maniru – omogućava istovremeno otvaranje više slika u odvojenim MDI prozorima (prozori za koje osnovni prozor programa predstavlja ceo ekran i van koga ne mogu izaći) Opcije za ručnu manipulaciju slikom se aktiviraju preko ikona koja su smašlene u odvojen prozor. pe so mogu po potrebi premeštali po ekranu Isti princip je primenjen i kod palele boja koje su takode u posebnom prozoru kao i kontrola za izbor veličine cetkice. Izbor komandi za ručnu manipulaciju je standardan (isecanje crtanje osnovnih geometrijskih figura i teksta brisanje sprej i popunjavanje zatvorenih kontura) uz svega par novosti kojima se u maniru speja, slika lokalno izoštrava ili čini mutnijom na nosi boja menja kontrast osvetljaj ili vrši stapanje boja. Simboli na ikonama jesu duhoviti i ilustrativni ali će vam u početku ipak više značiti kratko objašnjenje koja se pojavljuje u dnu prozora kada se kursor nađe na nekoj od ikona.

Ono što ćete varovatno više konstili su komande menija za globalnu manipulaciju sli-

kom. Sve su smeštene u meni **Edit** i grupisane u tri bloka: **filteri**, **slika** i **transformacije**. Filtera ima ukupno sedam i njima se slika iretrana isključivo tonski može promeniti na gotovo neograničen broj načina. Program će, recimo, pronaći nagle lokalne padova u intenzitetu i usaglasiti ih sa okolnom površinom čima zapravu uklanjamo sitne oštećenja sa slike. Zadati broj tačaka se može pretvoriti u površinu najpribližnije srednja tonske vrednosti, čime slika postaje mozaik.

Vrhunac složenosti je opcija **responsne curve** (kriva tonskih vrednosti) koju smo već pomenuli u osnovnom softveru za skeniranje. Ovde je ta opcija dovedena do savršenstva tako što se kriva može podešavati ne samo preko tri kontrolne tačke, već na niz načina uključujući i ručno crtanje. Kriva se može „razdvojiti" na boje i na svaku boju lokalno uticati.

Pod opcijom **image** se krije mogućnosti rastavljanja slike po RGB šemi na tri odvojene crno bele slike svaka u svom prozoru. Nakon manipulacije nad svakom pojedinačno slike se mogu ponovo spojiti u jednu. Od transformacija tu su klasična horizontalna ili vertikalna rotacija promene dimenzije slike, ali i pravljenje rubnog crteža (outline).

Već iz ovog nabrajanja koje nije potpuno stiće se sigurno utisak o mogućnostima programa. Ono što shvatite nakon kratkog vremena upotrebe programa je da proizvođač u stvari nije uspeo da odoli ign – slika u kojoj je svaka tačka predstavljena se 24 bita informacije omogućava da se primenom jednostavnih numeričkih operacija dobijaju najrazličiji efekti. Treba smanjiti intezitet zelene boje u senkama? Za računar zaista jednostavan zadatak – treba projuriti kroz niz bajtova koji predstavljaju zelenu boju i vrednosti u opsegu recimo 20 do 100, umanjiti za 10%!

Ostaje, naravno, pitanje da li su sve ugrađena transformacije korisniku zaista i potrebne odnosno da li će ikada biti upotrebljena? Odgovor bi mogao biti da! Zaista se može zamisliti situacija u kojoj će korisnik osetiti potrebu za tako delikatnim transformacijama skeniranog uzorka. Ipak, jedan drugi faktor praktičnu upotrebljivost ovih opcija prilično dovodi u pitanje, a to je ekran. Mogućnosti standardnog VGA podsistema da prikaže sliku sa ovom količinom informacije su prilično ograničene, pa se umesto boja na ekranu koriste tehnika rastera – nekoliko tačaka odgovarajućih kombinacija je boja zamenjuje stvarnu boju. Ekran time postaje dovoljan samo za manipulaciju slikom kao površinom – efekti lillera treba da budu izuzetno jaki da bi se na ekranu primetila razlika, pa time suptilna podešavanja koje su najčešće potrebna ako sliku treba kasnije reprodukovati na nekom kvalitetnom uređaju praktično otpadaju. Ipak, zbog ovoga ne treba previše brinuti – i sama 24-bitna kolor koncepcija je u ovom trenutku ispred mogućnosti PC hardvera, pa je na neki način i zadovljstvo videti program koji nagoveštava neka buduća, sigurno ne daleka, vremena.

Konačno pitanje koje se obično postavlja kada se srelnemo sa *Windows* programima je brzina. *Publisher's Paintbrush* se ovde više nego dobro snalazi i ogrnićen je prakticno brzinom samih *Windows*-a. Slika od 340K se učita za 3 sekunde, dok je polom potrebno još 7 sekundi da bi se konačno i pojavila u svom prozoru. Najveći broj transformacija koje smo pomiinjali se nad slikom te veličine odavi za per sekundi – najduže traje **outline** kome je potrebno oko 12 sekundi (sva merenje na 386 33Mhz, 64K cache, 4MB RAM). Za one koji su navikli na *Windows* okruženje program će svekako, biti dovoljno brz.

KVALITET CELINE

Skener i prateći softver, baz daljeg, predstavljaju retko atraktivnu celinu, koja u svakom trenutku odaje utisak već poslovičnog kvaliteta i preciznosti HP proizvoda. Naravno i pored sve atraktivnosti, osnovno je pitanje kolika je primenjivost ovoga uređaja? Odgovor na ovo pitanje je pokazao rezultate preko naših očekivanja!

Prvo što smo proverili je koliki je praktičan značaj povećane rezolucije. Znajući na osnovu iskustva sa prethodnim modelom da postoje crteži sa finim linijama koje jednostavno ne prolaze potunli smo ovom skeneru jedan uzorak koji je trebao da ga zbuni. Jedna šema formata 5 x 3 cm za zaista finim linijama se na ekranu pojavila se olprilike onom količinom informacija na koju smo i navikli – samo mali deo linija je bio vidljiv. Posle izmenađenja ovakvo standardnim rezultatima koje smo već videli kod sterog modela sa 300 x 300 dpi rezolucijom, izebrali smo opciju **izoštravanje** kod skeniranja i gle čuda – sve linije su pojavile! Naravno greška je učinjena sa naše strane, jer je skeniranje bilo obavljeno se korekcijama predviđenim za laserski štampač 300×300. Binarna slika je očigledno u sebi imala sve potrebne detalje da pruži mogucnost opciji izoštravanja da ih tako pojača da se oni pojeve i na izlaznom uređaju sa nešto manjom razolucijom. Ukratko, povećana rezolucija će se svakako osetiti na vašim crtežima čak i ako izlazni uređej ne može da podrži maksimum koji je skener u stanju da da.

Drugi aspekt upotrebe skenera koji nas je posebno zenimeo je kolika je upotrebljivost ovog skenera u procesu separacije boja kao pripreme za kolor štampu. Onima koji su upoznati sa načinom pripreme kolor štampe svakako je poznato da se profesionalni skeneri izuzetno skupi uređaji i da kolor separacije zahteva prilično znanja i preciznosti. Da li nešto što košta oko 2 500 USD može da zameni jedan ovakav uređaj čijim se cenama odično dodaju bar još dve nule?

Da bismo odgovorili na ovo pitanje preduzeli smo vrlo opsežna lastranja koja uključuju niz uređaja i programa – od samog HP skenera, preko laserskog štampača sa povecanom rezolucijom 600×600 i 800×600 do Linotronic 300 foto sloga se RIP-om u rezoluciji od 2540 dpi. Od programske opreme je konščen, porad pomenuta dva programa koji se dodijaju uz skener, i *CorelDraw!*. Za uporedni test, koji je obavljen u firmi Studio B&Z koršćeni su mekintoš računari (Apole Macintosh II Si) sa odgovarajućom programskom opremom (razdvajanje boja je obavljeno u *Adode Photoshopu*-u a štamparo iz *Quark Express*-a na Linotronic-u 330). Rađeno je sa TIF i PostScript formatima a rezultati su uporađivani sa onima koje daje profesionalni skener kojim raspolaže BIGZ-ova priprema. Oko procene vrednosti dobijenih rezultata smo konsultovali i profesionalce, kojima je ovo svakodnevni posao. Za sledeći broj pripremamo iscrpan dnevnik ovog testiranja uz niz ilustracija koje će vam predstaviti rezultate testa. Do tada, ostaje da samo zagolicamo vašu maštu izjavom da su dobijeni rezultati u najmanju ruku uzbudljivi!

Korisna adresa

Perihard
Ivana Milutinovića 24, 11000 Beograd
Tel: (011) 436-019, Fax: (011) 435 513

Štampači

LASER U KOLORU

"Tektronixov" novi laserski štampač nazvan „Phaser III PXi" po mnogo čemu odskače od ostalih uređaja te klase. Kao prvo, koristi naprednu ink-jet tehnologiju, nazvanu *phase change* koja omogućava štampanje na papiru i drugim podlogama u raznim veličinama i debljinama. Drugo, otisak rezolucije 300 dpi u punom opsegu boja odskače od otisaka drugih štampača delom i zahvaljujući podršci *Adobe PostScript Level 2*. Konačno „Phaser III PXi" odskače i cenom od $9995 što je vrlo povoljno u odnosu na ostale uređaje ove klase i kvalitet otiska.

Štampač je bazirran na AMD-ovom 2900 RISC procesoru koji radi na 24 MHz i sadrzi *PostScript Level 2* interpreter i Pantone-kompatibilnu simulaciju boja a pored toga implemantiran je i HPGL (Hewlett-Packard Graphics Language). U standardnu konfiguraciju spadaju 10 MB RAM-a (proširivo do 18 MB) i ROM-ovi koji sadrže 39 fontova uključujući Courier, Helvetica, Times, New Century Schoolbook i ITC Bookman.

Glavne zasluge za brljantnu kolor-grafiku pripadaju novoj ink-jet tehnologiji koja koristi specijalno netoksično čvrsto mastilo. Reč je o boji na bazi voska u obliku velikih krejona koji su oblikovani tako da ne može doći do greške u ulaganju u nosač (svako) boji odgovara poseban oblik). U toku rada boje se lope na temperatun od 140 stepeni – prilikom štampanja glave štampača raspršuje na papir kapljice istopljenog mastila koje se na sobnoj temperaturi momentalno hlade a dve valjka zatim utiskuju ohlađenu boju u površinski sloj papira.

'Phaser III' je opremljen hrpom I/O portova i indikatora – tu su mini-DIN 8 LocalTalk port, DB-25 serijski port, Centronics i 50-pinski SCSI port, namenjen hard disku za smestanje fontova. Kontrolni panel je opremljen malim LCD ekranom koji obavlja više funkcija – tokom štampanja prikazuje trenutnu aktivnost obaveštava o mogucim greškama i količini mastila svake boje u rezervoaru.

Priprema štampača za štampu je jednostavna, ali i relativno dugotrajna. Štampač se uključi, čelni krejona za boje (crna plava, žuta i crvena) se ubace u odgovarajuća ležišta, zatvore vratašca čima se krejoni poliskuju u rezervoar i to je sve – treba još samo sačekati 15-20 minuta da se štampač zagreje i mastile istope. Ovakva procedura znači da se štampač najverovatnije neće isključivati do kraja radnog dana, pa je zato predviđen i *standby* mod koji se uključuje posle dva sata neaktivnosti. Buđenje iz ovakvog moda je takođe automatsko – dovoljno je samo izdati nalog za štampanje – i traje oko 2,5 minuta a obuhvata zagravanje mastila i pročišćavanje glave.

"Phaser III" može da štampa na praktično svakoj vrsti papira koje postoji na tržištu u formatima do 11 x 17 inča sa ugrađenih 10 MB RAM-a, a sa dodatnih 4 MB format rastera do A3 (297 x 420 mm). Standardni ulagač papira radi sa američkim formatom A (8,5 x 11 inča) i DIN A4 (210 x 297 mm), formatom a opcioni univerzalni dodavač radi sa praktično svim američkim i evropskim formatima do formata A3.

Brzina je, na žalost, direktno proporcionalna kvalitetu – za štampanje pune A4 strane je u standardnom modu potrebno 2,5 min u kvalitetnijem (enhanced) načinu 4, a u najkvalitetnijem (premium) punih 7 minuta.

(M K)

DŽINOVI I PATULJCI

Evo i trećeg nastavka zbirnog pregleda ponude na domaćem tržištu. Uz svo poštovanje prema svim dosadašnjim i budućim učasnicima, pred nama je ovoga puta zsista jedna veličina. Reč je o modelu „Compaq Systempro" – sa dva i486 procesora i inteligentnim periferijama, on jednostavno nema konkurencije u mrežnom radu. A za one kojima ovakva silna snaga ne trebs, tu su, srećom, i druge, čini nam se, interesantne mašine.

Dejan V. Veselinović

Microsys

"Microsys" iz Novog Sada je trenutno jedini ovlašćeni prodavec računara firme „Compaq" koji nam je dostupan. Na sreću, imali su ne raspolaganju dve mašine, uz obećanje da će uskoro iza Nove godine biti i novih mašina u širem izboru. Na to možemo samo dodati – jedva čekamo! Nove mašine su osetno jeftinije od dosedašnjih modela „Compaq" a, koji su svi od reda bili u „Rolls-Royce" klasi (sa odgovarajućim cenama).

COMPAQ SYSTEMPRO

Procesor Intel 80486DX 33 MHz
Memorije 10 MB
Disk Compeq (420 MB)
Video Compaq VGA, VGA monitor

O ovom računaru je već bilo reči na stranicama časopisa pa bismo samo ukratko rezimirali. Radi se o mašini koje koristi EISA sabirnicu, mahom inteligentne periferale i jedan ili dva procesora. Na ovom nivou performansi, „Compaq" je smatrao da ISA sabirnice više neme smisla, pa sa odmah opredelio za EISA sabirnicu. Osnov logičke podrške čine dva već dobro poznata „intel" čipa (82357 i 82356), u okruženju se nekoliko „Compaq" VLSI čipova.

Kada kažemo da su periferali inteligentni mi to mislimo krajnje doslovno. Recimo disk kontroler kartica konsti procesor 80C186 (integrisane verzije starijeg procesora 8086) ne 16 MHz. Kartica za 8 terminele konsti isli takav procesor takode na 16 MHz. Dakle, ne računajući video čip u mašini postoje čak tri procesora, mada je samo jedan „zver" Intel 80486DX na 33 MHz. Delom dogradnjom računara možete ugraditi još jeden takav isti procesor (ili neki slabiji, recimo 80386 na 33 MHz – mada bi to bio greh!).

Međutim, da biste do kraje iskoristili ovakvu mašinu, prosto je šteta koristiti je za običan DOS iako to nerevno jeste moguće. Istina, malo je verovatno da bi iko, ma kako bogat, kupio ovaj model za obične DOS aplikacije – ovo je namenjeno ulozi servera. Sa tim u vidu uz računer dobijate i mnogo disketa sa raznim potrebnim programima. Recimo, podešavanje EISA sabirnice i opšte konfiguracije podrške za OS/2, podrške za *Windows*, podrške sa *Novell Netware* itd. Tek uz pomoć ovih programe i pažljivog planiranja i konfiguracije možete reći da ste ga iskoristili.

Elektronika je, blago rečeno, „superlativna". Na celoj matičnoj ploči, samo su veznici (video, serijski i paralelni) i stabilizatori napone zalemljeni klasičnom tehnologiyom, sve ostalo je površinski lem što važi i za sve pločice sem poslolje za razne čipove. Ono što se odmah može uočiti je zaista veliki broj VLSI kola sa oznakom „Compaq" (vrlo pohvalno, sami razvijaju potrebnu veznu logiku – skupo ali dobro). Sistem sadrži ukupno osam ISA/EISA utičnica za proširenje, dve posebne utičnice za sistemsku memoriju, jednu posebnu utičnicu za procasorsku karticu i jednu utičnicu koje može zadržati ili procesorsku ili memorijsku karticu.

Imajući sve to ovo u vidu osećali smo se pomalo glupo kada smo ovaj računar testirali standardnim DOS testovima, onako samog za sebe i da smo imali drugi procesor ništa ne bi vredelo, jer običan DOS ne može da ga iskoristi. Sa diskom od 420 MB i 10 MB sistemske memorije ovo očigledno nije namenjeno konsnicima pojedincima a to je tek početak u ovakvog kakav je možete ugraditi još tri takve diske do 256 MB memorije, drugi procesor još 4-5 kartica za terminele – ukratko sistem za stotinak konsnika „No problemo!". No dužnost je dužnosl pa smo propustili sve uobičajene testove.



Za one koji cene kvalitet: DeskPro 386s/20



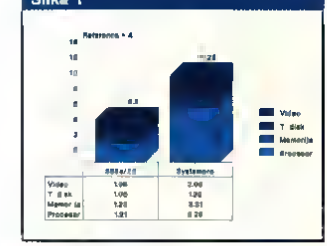
Bez konkurencije na najtežim poslovima: Compeq SyetemPro 486DX

Iz priloženih rezultata se lako može videti da je ovaj sistem na vrhu lestvice brzine mada uz čisto akademsku razliku u odnosu na druge, ovaj princip da „Compaq" više ne zauzima ubedljivo prvo mesto po performansama, postaje prilično uobičajen i u programskim i u mašinskim testovima, ima drugih koji su goto jednako brzi ali i jeftiniji, ieko epsolutno gledano nisu ni malo jeftini.

No ponevijemo, sa već dvadesetak (a kemoli više) konsnika povezanih u mrežu u kojoj je ovej računer sarver od konkurencije (bar ne našem tržišlu) neće ostati ni „k".

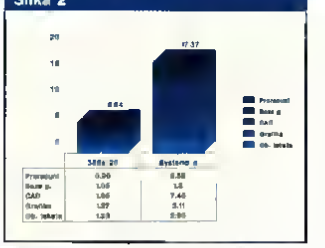
Divili smo mu se kada se pojavio vremenom, pojavili su se drugi koji su brzi I jeftiniji ali u tehničkom, e posebno u funkcionalnom

Slika 1



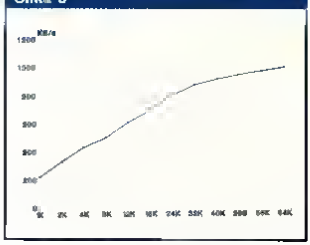
Mašineki

Slika 2



i progremski test računara Compaq

Slika 3



Brzina prenosa tvrdog dieka (SystemPro)

smislu. „Systempro" i dalje suvereno vlada. Ovo zapravo i nije lični računar ovo je maskirani mini koji se sprama (sa novim procesorima na 50 MHz i većim diskovima, kao i sa dodatnim nizovima diskova sve do 20 GB) de zagrize i klasu midi mašina.

Kao sadašnji mini, a sutra ako treba i midi, bez konkurencije je i – fenomenalan!

COMPAQ DESKPRO 386s/20

Procesor Intel 80486SX 20 MHz
Memorija 16 KB keš, 2 MB
Disk Compaq (60 MB)
Video Compaq VGA, VGA monitor

Sa ovim računarom, odnosno njegovim prethodnikom na 16 MHz, smo već imali posla, mada poodavno. U međuvremenu, „Compaq" je podigao radni takt za 25% (sa 16 na 20 MHz) a koliko je nama pozneto bio je prvi koji je na tržište izbacio računar sa ovim procesorom i statičkom keš memorijom. Naravno, od onda do sada, obe stvari su postale standerd, a već se prelazi i na 25 MHz, opet 25% brže. Zato nas je zanimalo da pogledamo kako danas izgledaju odnosi u ovom segmentu tržišta, na samom pragu svela 32-bitnih mašina.

Kućište mašine je sada već standardnog „baby AT" formata koji „Compaq" konsti. Kao i uvek od ove firme, mehanički je sjajno zamišljeno, izvedeno i obrađeno. Isti princip važi i za sve ostalo: flopi disk jedinicu je potpisala japanska firma „Citizen", disk je nepoznatog porekla (najverovatnije „Conner"). Video konsti standardnu VGA rezoluciju i inače odličan monitor koji proizvodi (ili bar potpisuje) sam „Compaq".

Tastature je uobičajeni „Compaq"-ov model. I dalje ostaje maltene usamljena po reskuji konsinika – nju ili obožavate, ili mrzite, neme sredine. A neki aspekti su zaista pomalo smešni, recimo. „Compaq" je valjda poslednja firma na svetu koja neme taster „Enter" u obliku obrnutog slova „L", već kao produženi horizontalni taster. Valjde se vremenom naviknete, ali ipak.

Po performansama poredeći ovu mašinu sa ostalim sličnim modelima, videćete da se ovaj model, uprkos vremenu, veoma dobro drži. Zahvaljujući odličnoj integraciji, a svakako i keš memoriji jedva da je nešto malo sporiji od sabraće koja rade na 25% većim taktovima. Odličan video podsistem, tradicionalno jaka strana „Compaq" računara, pomogao je ovoj mašini da postigne indeks od 1,37 na hardverskim testovima, dakle, čak 5% brže od „Imtelovog" SX-a na 25 MHz.

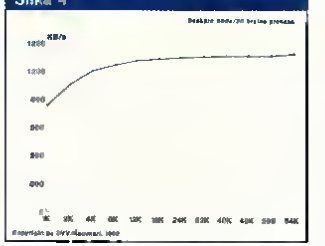
Na programskim testovima, 386s/20 je postigao indeks od 1,17, a pošto je radio pod DOS-om 4.01 („Compaq"-ova licenca) nije teško pretpostaviti da bi ukupan rezultat bio nešto bolji da je radio pod verzijom 5.0. Probali smo i to i zaista, prosečni indeks je porastao na 1,21, čime se opet približio sličnim mašinama na 25 MHz. Prema tome, ovo je (i dalje) najbrža SX mašina na 20 MHz, koja je praktično iste brzine kao i formalno brže mašine u toj klasi.

Ova bombona od mašine bi ionako dobila preporuku, čak i uprkos ceni za koju smo čekivali da bude vrtoglavo velika, ne naše prijetno iznenađenje, saznali smo da cena, iako ne mala, nije ni onolika koliku smo očekivali. Nekada su „Compaq" mašine koštale koliko one od najskupljih klonera u gradu pa puta tri, a danas su to cene najskupljih klonera puta faktor manji od dva, a mnogo bliži 1,5. Sa tim u vidu, preporuka zaista mora da preraste u veoma jaku preporuku – ako cenite kvalitet obavezno se raspitajte kod „Microsysa" pre nego što išta kupite.

Korisne adrese

MS Microsys
21000 Novi Sad
Tel. (021) 611-366, Fax. (021) 20-544
Poslovnica Beograd (011) 622-581

Slika 4



Brzine prenosa tvrdog diska (Compeq 386)

IMTEL

"Imtel" (skraćenica od „Institut za mikrotalasnu tehniku i elektroniku") dugo se bavio razvojem posebnih uređaja i njihovom proizvodnjom u malim serijama. Ne tako davno počeo je da se bavi i prodajom ličih računara, po čemu su se dosta brzo pročuo zahvaljujući veoma niskim cenama. Te cene je postigao nudeći opremu na kraći „poček", a kome se žurilo, mogao je da plati malo više i dobije opremu odmah. Mi smo se uverili da je imaju na legeru, porazgovarali smo u ponedeljak a opremu dobili u utorak, na samu Novu godinu. mi bismo ovo nazvali promptnim, a to je na tržištu uvek vrlina.

IMTEL 286

Procesor Intel/Harris 80C286, 16 MHz
Memorija 1 MB
Disk Seagate ST-157A (42 MB)
Video Leadman VGA (mono)

Iako se radi o već relativno zastarelo tehnologiji, ovaj model nastavlja da se prodaje veoma dobro. A zašto i ne bi, kada je tako pogoden za sekretarske, manje službeničke poslove i, nerevno, kao nod u mrežama.

Primenjena tehnologije bi se mogla nazvati ultra klasičnom. Matična ploča koristi „Headland" G2 HT12 čip koji objedinjuje maltene sve funkcije AT mašine. Memorija se može montirati u obliku DIP čipova (prvi megabajt) ili pak u obliku četin SIM modula, ovo je mala izmena u odnosu na starije ploče ovog tipe koje su koristile SIP module, izmena koju smatramo korakom napred. Mi smo dobili ploču sa 1 MB u DIP čipovima.

Flopi disk jedinicu od 5.25 inča potpisale je japanska firma „Chinon", a disk firma Seagate. Radi se o sada već klasičnom modelu ST-157A ili njegovom nasledniku ST-357A (u zavisnosti od raspoloživosti, jer trenutno vlada nestašica tvrdih diskova na svetskom tržištu). Jedine stvar koja nije uobičajena je opredeljenje „Imtela" na VGA grafiku i to uz svojevrsnu standardizaciju. Naime „imtel" standerdno nudi „Trident" TVGA video karticu sa „Trident" 8900 video čipom, pri čemu se memorija montira na postolje, pa je lako možete podići sa 256 prvo na 512 KB, a potom i na celih 1 MB (ovo poslednje vam treba samo za režim od 1024×768 se 256 boja). Za mono verziju koju smo mi dobili, sasvim je dovoljno i 256 KB (ali godi da znate da ako predete na boju, ne morate menjati karticu, već samo dopiatiti dva čipa).

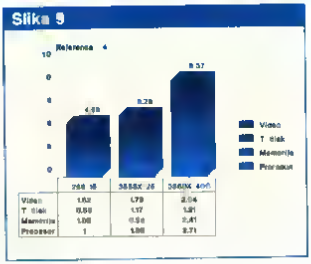
Monitor koji uz ovo ide je generički proizvod koji amo do sada videli pod bar tri imena u ovom slučaju, zvao se „Leadman" model LM-7260. Radi se o modelu koji koristi P-219 fosfor (nizak stepen zračenja) u standardnoj VGA rezoluciji od 640×480, isto tako, pali od potpuno istih mana svoje generičke sabraće, a to je malo iskošena slika (nije savršeno horizontelna) uz primetna izobličenja po ivicama u odnosu na centar ekrana. Sve ovo nije katastrofalno, ali jeste danak ceni.

Kućište spada u grupu onih koja se lako otvaraju – zgodno ze nas koji stalno nešto petljamo po mašini, ali ne valja za radne organizacije. Iak pristup ohrabruje ljude da nešto petljaju. Po nama, ovo je kutija sa najneprivlačnijom maskom koju smo do sada videli, ali to je nesravno stvar ukusa. Tastatura je YU verzija prilično odomaćene firme „Chicony", model KB-5191. Kao plus, navodimo da „imtel" uz ovaj monitor i ovu tastaturu daje i disketu koja pre-

bacuje i jedno i drugo na YU standarde (delalj koji nije ni malo modni a koga tako često nema).

Na mašinskim testovima pre svega zahvaljujući upotrebi DOS-a 5.0 i relativno brze video kartice, ovaj računar je postigao indeks od 1.15, podsetimo da i naša referenca koristi sličnu tehnologiju, ali sa DOS-om 3.3 i nešto sporijom video karticom (iakođe VGA). U programskim testovima postignuti indeks iznosi 0.96, a razlika potiče pre svega od činjenice da naša referenca ima numerički koprocesor, a ovaj modal ga nije imao (naravno – to je uvek opciona stvar). Kada se ukloni koprocesor, indeks raste na 1.02, a bio bi i veći da ga ne kvare rezultati testa sa bazama podataka sa indeksom od 0.90.

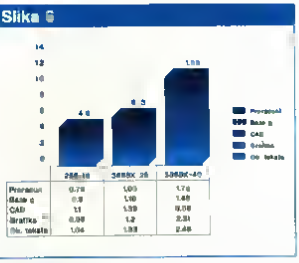
Slika 5



	286-16	386SX-25	386DX-40
Video	1,52	1,79	2,04
T. disk	0,99	1,17	1,31
Memorija	1,00	0,96	2,41
Procesor	1	1,06	2,71

Mašinski...

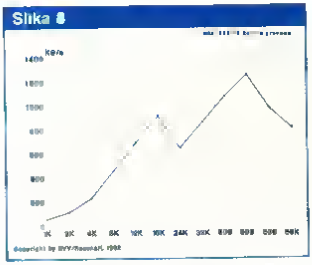
Slika 6



	286-16	386SX-25	386DX-40
Procesor	0,78	1,03	1,78
Base 6	0,9	1,16	1,48
CAD	1,1	1,39	3,08
Grafika	0,98	1,2	2,81
Dio. teksta	1,04	1,33	2,48

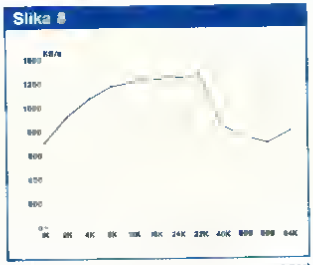
...i programski test računara Imtel

Slika 4



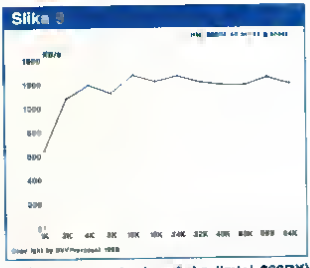
Brzina prenosa tvrdog diska (Imtel 286/16)

Slika 8



Brzina prenosa tvrdog diska (Imtel 366SX)

Slika 9



Brzina prenosa tvrdog diska (Imtel 386DX)

Interesantno je primetiti da naša referenca koristi Quantum LPS52A isti onaj kao u velikoj „Imtel" mašini i isti onaj koji je ova firma sladodarno isporucivala pre neslašice i prekida slabijih kanala nabavke (zaslupnik je bila ljubljanska firma ATR).

Sve u svemu, radi se o sasvim standardnoj mašini današnjice koju od konkurencije odvaja samo VGA grafika i prihvatljiva cena. Ako vas interesuje zahtevajte malo bolje pa makar i malo skuplja kućišta, uz tu napomenu, slobodno ga uzmite u obzir ne možete pogrešiti.

IMTEL 386SX

Procesor Intel/AMD 80386SX 25 MHz
Memorija 2 MB
Disk Quantum LPS52A (52 MB)
Video Leadman VGA (mono)

Evo i prve komercijalne mašine u klasi SX procesora na radnom taktu od 25 MHz koju vidimo na testu sistu. Uskoro će većina prodavaca krenuti ovim putem – SX na ovom taktu predstavlja razumnu zamenu za skuplji DX na istom taktu i bez keš memorije, a po relativnoj računarskoj snazi, ova platforma može da zadovolji calu srednju klasu korsnika.

Polazeći od relativne mane prethodnog modala kućišta, ovaj racunar je bio zapakovan u daleko bolje (i estetski prihvatljivija). ali nasumnjivo i mehanički čvršće kucište (sa poprečnim učvršćivačima iznutra). Sa prednje strane kucište nudi tri dostupne jedinice od 5.25 inča i jednu od 3.5 inča, mada iznutra imala mogućnosi monlaže još dve spolja nevidljive jedinice od 3.5 inče. Pošto ona imaju trecinsku visinu, možete upotrebiti dve za recimo dopunski tvrdi disk – napajanje vam nece biti problem, jer nudi 200 VA, a jedino vam može zasmetati to što biste se u tom slučaju morali odreci puna duzine ulicnica. Pošlo je danas malo kartica koje traže punu duzinu praostale dva ulicnica bi vas verovatno zadovoljile.

Druga razlika u odnosu na bazni model je tvrdi disk. U ovom slučaju bio je to jedan „Quantum" modal LPS52A, formatiranog kapaciteta od 52 MB. Ovaj disk je i naš relarentni disk kada se pojavio, bio je apsolutno izuzetan, a danas, nakon godinu i više dana, ostao je na samom vrhu svoje klase (mada više nije onako usamljen). Bio je i ostao odličan disk.

Najzad sama matična ploča. Interesantno je zapaziti da je napravljena na na Tajvanu, već u Hong Kongu. Rad je odličan a slično prvoj takvoj ploči o kojoj smo ranije govorili i ova sadrži niz „diskutabilnih" detalja. Prvo nudi šest ulicnica za proširenja (4 16-bitne i 2 8-bitne) što je malo obzirom da sama matična ploča nama nišla na sabi, a nalik je prvom primerku koji smo probali. Drugo, opet nalik prvom primerku, ploča ima formal otprilike polovine „baby" AT ploča, isto kao i G2 ploča – ovako male dimenzije su uslovile odsustvo nekih stvari koja su mogle da se slave na ploču bez ikakvih problema, kao racimo AT Bus kontrolar i sve ulazne i izlazne linije (kao na sada dve i po godina staroj DEICO SX ploči). Tima bi cena porasla nekoliko procenala, ali bi porasla i praktičnosi ploče šime što bi se nadoknadila bar jedna ulicnica za proširenja. Najzad koncepcija rasporeda memorije na ploči je malo cudna – kombinuje DIP i SIM čipove u neobičnoj konfiguraciji, mada ploča prima do 20 MB memorije više nego što će verovalno ikada biti na nju stavljeno.

Racunar je isporučen sa 2 MB mamorije u vidu SIM modula, uz obrazlozenje da je ovo najčešca konfiguracija. To možda jeste tako, ali dva modula čine jednu banku, a to znači da procesor ne može da prepliće, čime se dobija veća brzina rada. Ovo je i mala preporuka – ovaj racunar uzmite ili sa 1 MB (4×256 KB SIM) ili sa 4 MB (4×1MB SIM).

Razlika između ove i prve ploče koju smo

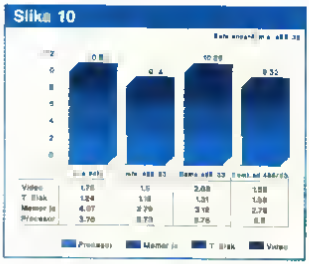


Ako vam treba jaka maalna: Imtel 386DX

probali je i u važnoj logici. Prva je korišila OPTI čipove, a ova koristi čipove firme VLSI (82C320 i 82C331) i BIOS potpisan od firme „Quadtel", poznate sa DEICO SX ploče (R/Optima SX). Nemamo funkcionalnih zamerki na BIOS, ali MR BIOS ostaje pojam odbio pogodanog niza menija i opcija na njima.

Ovo je dakle platforma namenjena srednjoj klasi, sa ciljem da zameni DX mašine (skuplje a samo malo brže, par procenala) i da zadovolji srednju klasu korsnika. Obzirom na mogućnosti širenja, ako nemala prevelike zahteva, ovaj računar bi mogao da vas poknje sada i u skorijoj budućnosti, imajte ga u vidu.

Slika 10



Mašinski...

IMTEL 386DX

Procesor Intel/AMD 80386DX 40 MHz
Memorija 64 KB keš 4 MB
Disk Maxtor 7120-AT (130 MB)
Video Leadman VGA (boja)

Ova platforma je sligla u jednom „mini tower" kućištu dubiozne estetike ali neosporno kvaliteta, primera radi, ceo unutrašnji skelet kućišta je pocinkovan i to sa veoma visokim kvalitetom finalne brade. Konstrukcija spada medu najbolje do sada viđana u ovoj klasi.

Tvrdi disk je u ovom slučaju bio jedan „Maxtor" 7120 AT formatiranog kapaciteta od 130 MB. Flopi diskelna jedinica je opet od firme „Chinon", a video karta koristi „Trident" TVGA 8900 video čip. U ovom modelu kartica je sadržala ceo magabajt video memorije monitor je opet od firme „Leadman", ali sa oznakom LM-7300 i za razliku od manjeg brata ovaj je u boji. Slika je vidno kvalitetnije od jednobojnog modela, sa prihvatljivo malim ugaonim izobličenjima, sasvim pristojnim bojama i malim stepenom elektromagnatskog značenja.

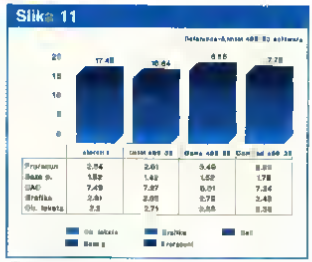
Matična ploča koristi procesor AMD 80386DX na radnom taktu od celih 40 MHz. Naravno tu je i keš memorija u ovom slučaju 64 KB koja se sastojala od brzih SRAM čipove (20 nS) i još bržih TAG RAM čipova (15 nS). Vezna logika se sastojala od nama do sada potpuno nepoznatih „Forex" čipova (4L04F1244 i 4L05F2243), potpomognutih jednim IMP čipom (80C206P) koji se starao o sabirnici i spoljnoj memoriji. Ploča prima u raznim kombinacijama do 32 MB, a tu je posebna ulična za još jednom toliko 64 MB u mašini već nije ni malo za šalu posebno ako računar radi pod UNIX-om ili nekim drugim upravljačkim sistemom koji koristi linearan pristup resursima a pre svega memoriji.

Sasvim očekivano ovaj model je sebi obezbedio odlične rezultate. Na mašinskim testovima, postigao je indeks od 2,09. Što se sasvim lepo poredi sa recimo indeksom od 2,81 za „Compaq Systempro". Interesantno je napomenuti da su indeksi za tvrdi disk ovog računara (1.21) i pomenuta „Compaq" mašine (1.22) praktično sasvim isti.

Na programskim testovima ovaj model je postigao indeks od 2,21, što je veoma dobro. Posebno ističemo rezultat testova sa bazama podataka gde je postignuti indeks iznosio čak 1,48. Sve u svemu ovaj računar je oko dva puta brži od manjeg modela sa procesorom SX.

Smatramo da ovu platformu treba shvatiti kao 486 za siromašnije ili one kojima ne trebaju usluge numeričkog koprocesora. Ako vam traba numerika 486 je bez premca a ako vam to nije nužno ili uopšte potrebno ova platforma je veoma realna alternativa ako vam naprosto

Slika 11



...i programski test računara Imtel 486DX



Imtel 386SX/25

treba jaka mašina razmotrite ovaj model čak i kao server u manjim mrežama (ali sa većim diskom).

IMTEL 486DX

Procesor Intel 80486DX 33 MHz
Memorija 256 KB keš 4 MB
Disk Maxtor 7120-AT (130 MB)
Video Trident 8900 VGA (boja)

I najzad evo najvišeg dela ponude firme "Imtel". U pitanju je naravno, računar na bazi matične ploče sa procesorom Intel 80486D na 33 MHz i sa velikodušnih 256 KB keš memorije. U celini, radi se o prilično tipičnoj mašini u ovoj klasi.

Proizvođač ploče se potpisao samo imenom modela ploče, navedenim kao "CO486DX/33", što smo saznali iz propratne odkumentacija. Ta dokumentacija je zapravo, jedna, knjižica koja spada u red osrednje informativnih od pluseva navodimo veoma informativan deo pnrućnika u vezi kraikospojnika, memorije i slično, a od minusa opise ekrana programa u ROM-u za podešavanje koji ne zadovoljavaju po kvalitetu opisa.

Same ploča je negde izmedu veoma dobrog i odličnog rad je odličan delovi (utičnice postolje ze memoriju) odlični a raspored kratkospojnika pristojan. Program za podešavanje je potpisala firma AMI, a nije baš najsvežiji posledniji datum autoretva je iz 1990. godine. Ploče prima SIM module od 256 KB 1 i 4 MB sa ukupno 32 MB na samoj ploči i još toliko na posebnoj memonjskoj kartici sa 64 MB sistemske memorije mnogo loga poslaje moguce pod mrežama i nekim upravljačkim sistemima.

Video kartica koristi Todeni 8900 video čip sa 1 MB video memorije, što znači da može da radi do rezolucije od 1024x768 sa ili bez preplitanja i sa 256 boja. Na žalost monitor pod imenom "Leadman LM-7300" iako spada u klasu sasvim pristojnih monitora ovog tipa (dijagonala ekrana 14 inča, veličina lečke 0.28 mm), ne može više od 1024x768 sa preplitanjem. Tastatura je standardan "Chicony" KB-5191 model ni dobar ni loš ali sa našim znacima, napominjemo da "Imtel" standardno isporučuje i disketu sa onim malim ali prečesto zaboravljanim pobudnim programom za nase znake, kako bi oznake na tastaturi bile verno prikazane i na ekranu. Sitnica ali veoma korisna sitnica.

Tvrdi disk je potpisala firma "Maxtor", e kapacitet je sasvim solidnih 130 MB formatirano. Kućište spada u "mini tower" senju. nije baš estetski doživljaj, ali ima jednu interesantnu i veoma korisnu sitnicu, koja ga cini boljim od proseka. Pored dva standardna mesta za jednice od 3.5 inča sa trećinskom visinom koje su pristupačne spolja postoji i treće mesto za tu kvu jedinicu iznutra, koje nije pristupačno spolja ali zato ima nešto veću visinu odnosno poluvisinsko ja. Očigledno je namenjeno ugradnji tvrdih diskova čak i onih malo većih a to znači kapacitala sve do 425 MB. Ovo rešenje je dobro jer kada se disk dobro slegne, laj okvir pomaže odvođenju toplole čime postaja neka vrsta hladnjaka.

U toku rada, našli smo na samo jedan problem. I to u vezi sa video kartom. Prvo, karta je odbila da radi sa stanjom verzijom programa "Windows" uspul ruševi sistem. ovo je blago rečeno čudno pošto se radi o verziji programa za standardnu VGA rezoluciju. Drugi problem se pokazao u slučaju sprega monitor-kartica koji nisu mogli da se usaglase sa programom "Harvard Graphics" 2.31, razvlaceci ekran van polja vidnosti u slučaju monohromatskih VGA monitora. Opet, veoma čudan efekal koga ne bi trebalo da bude. Sve ovo navodimo kako bismo objasnili izuzetno mali indeks na mašinskim testovima video podsistema - jednostavno, morali smo da se oslonimo samo na merenja u tekstualnom režimu. Pošto su nam u Imtelu rekli da oni smatraju da ne treba uključivati dipikalnu memoriju (shadow RAM) za sistem (?) i video (???), moraće da se zadovolje otprilike polovinom indeksa koji su mogli da imaju.

U programskim testovima koja smatramo merodavnim, ovaj računar se u celini pokazao čak veoma dobrim. Naizgled na baš sjajan Maxtor disk je u sprezi sa DOS-om 5.0 dao odličan rezultat u svim slučajevima, a posebno u radu sa bazama podataka. Indeks brzine od 1.42 nije baš svakodnevna pojava ni u ovoj klasi. Opšti nivo rezultata je iznad proseka što je za pohvalu.

Sve u svemu, ovo je mašina koju ne treba olako odbaciti utoliko pre što se radi o modelu bližem dnu nego vrhu lestvice ponude matične firme. Bili bismo srećniji sa boljom video karlom, jer nam se nekako čini da platforma zaslužuje bolje. Imajte ga u vidu ako razmišljate o ovoj klasi.

Korisna adresa

IMTEL
11070 Novi Beograd
Bulevar Lenjina 165- b
Tel. (011) 135-420 Fax (011) 138-928

OBRAČUN KOD O.K. PIKSELA

Pet godina nakon pojave VGA standarda, na tržištu su se iskristalisala dva proizvođača video čipova koji zajedno podmiruju potrebe oko 65% tržišta. Njihove cene su različite, mogućnosti slične, pakovanja mnogobrojna... ali gde su u svemu tome korisnici? Pokušali smo da pronađemo odgovor na ovo pitanje.

Dejan V. Veselinović

Uglavnom je sviме jasno da su deni Hercules grafike odbrojani jedino što je još održava na tržištu je zaista skromna cena. Drugim rečima, Hercules grafike danas predstavlja najniži segment tržišta, nešto kao grafiku za siromašne. Nije stvar u tome da se radi o lošoj grafici, setimo se pre pet ili šest godina, to je bilo bolje od bilo čega što je IBM nudio u vidu standarda, sem ukoliko niste epsolutno insistirali na boji. Najzad, u to vreme, razlika u ceni između Hercules i EGA grafike je bila 5-8 puta u korist Herculesa.

NEZAVISNA PROIZVODNJA

Ali kao i uvek vremena se menjaju. EGA standard je zamenio VGA standard kada je aprila 1987. godine IBM prikazao svoju potpuno novu seriju računara pod nazivom PS/2. Od onde do danas, iako je VGA ostao poslednji opštevažeći standard, koji se tek sada transformiše u noviji i daleko bolji XGA standard, taj ulazni nivo je prevaziđen. Neravno prevazišle ga je konkurencija, koja je, da bi opstala morala da ponudi više za manje para od IBM-a. Prvo su došle veće rezolucije (800×600 i 1024×768), pa veći broj boja (256 umesto 16), a sada se dogurało do kombinacije velike rezolucije (1024×768) u saradnji sa velikim brojem boja (32 000 i više).

Da bi sve to uopšte bilo moguće, neko je morao da napravi video kola koja će u potpunosti imitirati rad IBM elektronike, ali tako da se izbegnu svi problemi oko autorskih i patentnih prava. Čitav niz firmi je taj posao udarnički obavio. „Chips & Technologies", „Peradise" (sada deo „Western Digitale"), „Oak", „Ahead", „Trident", „Tseng", „Genoa", ATI, „Video 7 (sada deo firme „Headland"), itd. Neke od pomenutih firmi su u međuvremenu, maltene nestale, neke rade isključivo za sebe, a neke su opet aktivne na svim frontovima. Nas zanimaju ove poslednje.

Proizvođači video čipova se u načelu dvojako ponašaju. Ili rade samo za sebe pod sopstvenim imenom, ili pak rade na tzv. OEM (Original Equipment Manufacturer – originalni proizvođač opreme, to jest „je pravim pa lepim tvoje ime, a ti prodaješ kao svoje") principu. To znači da se svi proizvodi prave u jednoj ili dve fabrike, da bi se nakon toga prodavali pod nizom različitih imena, a bogami i se veoma različitim cenama. Od poznatijih proizvođača video čipova koji oveko rade nevodimo firme „Ahead", „Oak", „Trident" i „Tseng". Prva i treća su američke po poreklu (mada se fabrike nalaze na Tajvanu), a druge i četvrta su u celosti tajvanske firme.

Od ove četiri firme, „Tseng Labs" i „Trident" su se nekako probile na vrh, afirmišući se kao liderи grupe. Prve su izbacivale nove proizvode, veoma predano radile na razvoju i tehnologije i video standarda i narevno prodavale karlice u što većim brojevima. Međutim, jedna razlika koja je između njih postojala od samog početka je ostala i do današnjih dana – cena.

„Trident" je uvek bio jeftiniji od „Tsenga" (navodno, Tajvanci to čitaju kao „Đeng"), ali je upravo to bio razlog da „Tseng" uvek bio nekako više poštovan od konkurencije.

Kod nas, nude se obe kartice, neretko i od istih prodavaca, mada ima i jasne polarizacije, uglavnom u korist firme „Trident". Ovo nije lesko objasniti jednom jednom rečju – cenom. No, bez obzira na sve to, naša iskustva sa „Trident" karticama nisu baš bila za pohvalu, ali su ona sada bret-bratu godinu i po dane stara, pa više ne moraju biti merodavna. Poslednji elemenat koji nas je naveo na ovaj projekat je diskusija o video karticama na Sezamu u konferenciji PC HARD. Postalo je jasno da se korisnici polarizuju oko video karti mallene kao oko programa za obradu teksta – bez milosti se brani svoje, neretko i ocrnjavanjem tuđeg.

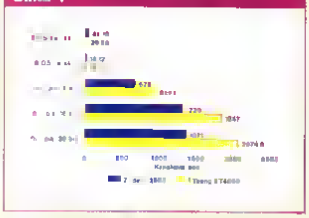
Sve, to zajedno, kao i objektivne razlike u ceni, navelo nas je da se ozbiljno pozabavimo upravo ovim dvema video karticama. Pošto smo se ograničili na svega dve, odlučili smo da ispitamo ne samo kartice u standardnom VGA režimu, već i u posebnim režimima. Tačno je da svi ovi posebni režimi podrazumevaju programske veznike, ali smetnimo da su i oni deo paketa koji na ovakvom tastu treba uzeti u obzir. Šta će vam karte bez veznika, a pošto su veznici programi, treba očekivati razlika i u njihovom kvalitetu. Najzad pošto velike rezolucije podrazumevaju ove programe, i vas i nas zanima kartica kao simbioza programa i hardvera, a ne kao odvojeni delovi.

ČISTA ELEKTRONIKA

Kartica sa „Tseng" ET4000 video čipom je tipičan generički proizvod za koji se tajvanski dobavljač nije potrudio čak ni da izmisli neko

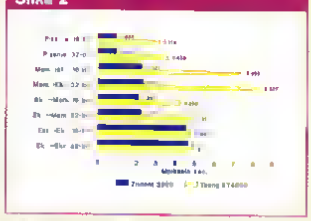


Slika 1



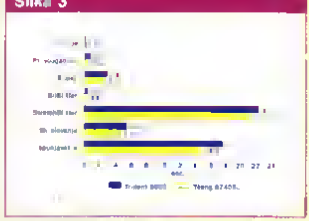
Trident vs Tseng: Brzina rada u tekstualnom režimu

Slika 2



Trident vs Tseng: Brzina rada u grafičkom režimu

Slika 3



Trident vs Tseng: Brzina rada pod *Windows* operativnim sistemom

zvučno ime ako se izuzme „Enhanced VGA" (poboljšana VGA). Štampana pločica, jasno pokazuje dokle su Tajvanci stigli sa razvojem ove tehnologije – do samog vrha. Rad i štampa su zaista odlični – prvoklasna izvedba.

Kartica je inače dosta velike za današnje pojmove, mada je ujedno i relativno prazna. Pored video čipa sa oznakom „Tseng" ET4000AX sadrži osam memorijskih čipova na postoljima, 5 TTL čipova, jedno PAL kolo, jedno kolo za generisanje potrebnih učestanosti ze sinhronizaciju, dva BIOS kola, DAC i jedno BIOS kolo ze prevođenje. Deluje mnogo ali zaista nije ni polovine površine kartice je prazna.

Autor BIOS-a je sam „Tseng". Memorija se sastoji od dva, četiri ili osam DRAM čipova tipa 4×256 KB (u ovom slučaju SAMSUNG KM44C256AP-8, 80 nS).

Izvedba „Trident" kartice je u najmanju ruku jednako dobra kao i kod „Tsenga", ako nije i za dlaku odlja. Kao prvo, kartica je fizički nešto manja, pa se približno istim brojem delova i opštom konfiguracijom deluje kompaktnije i popunjenija.

Pored očekivane razlike u video čipu „Trident" kartica ima BIOS koji je potpisala američka firma Quadtel, nama poznata od našeg modela R/Optima SX. Sasvim nalik konkurenciji, „Trident" kartica ima mogućnsi podržavanja do 1 MB video memorije, opet u DRAM čipovima u arhitekturi 4×256 KB. Za razliku od „Tseng" kartice, dva takva čipa su neposredno zalemljena na karticu, a za ostatak su obezbeđena podnožja. Kako one dva zalemljena, tako su i preostalih šest DRAM čipova imali oznake dosta brze memorije od 80 nS (mada iz dva izvora – zalemljene je potpisala firma „Micron Technology" a one u postoljima firma „Goldstar").

VEZNI PROGRAMI

Priručnik za „Tseng" karticu je takođe ga nančki proizvod, ali je štampan na veoma dobrom papiru, lepo je povezan i dosta je informativan. Sa jedne strane, sadrži informacije koje se retko viđaju (sistemski pozivi BIOS-a) a se druge strane, nije baš preterano informativan u kontekstu instalacije veznih programa.

Priručnik za „Trident" karticu je takođe dobro pripremljen i izveden, ali sa potpuno drugačijom koncepcijom. Ovaj priručnik programeru ne nudi praktično ništa, ali zato daje detaljna objašnjenja raznih pomoćnih programa koji se isporučuju uz karticu. Na žalost, uopšte nema reči o instalaciji veznih programa – sve se svodi napros to na jedan spisak šta je i u kojoj verziji podržano. Ovo je, po nama, ipak malo čudno, nekoliko reči ih ne bi ubilo.

Tabele u priručnicima prikazuju vezne programe koji se dobijaju uz kartice. Napominjemo da se veznik „Trident" kartice za Windows instalira u jednom jedinom prolazu, dok vas veznik za „Tseng" zaista maltretira do besnila (ukupno pet prolaza, nikada kraja disku ovom ili disku onom).

Druga značajnija razlika na koju smo naišli se odnosi na veznike za program *WordPerfect 5.1*. „Tseng" vam nudi samo visoku razolucuju u grafičkom režimu, dok vam „Trident" nudi i razne posebna oblika rada u tekstualnom režimu, sve do 132×50, ili 3 3 puta više teksta na ekranu od standarda. Iz iskustva znamo da nam je ovako nešto odavno trebalo i na „Tseng" karticama, ali toga još uvek nema. Vi to možete poslići eksterno, pozivajući opšti program ze podešavanje režima rada, ali ne i interno, kao kod „Trident" karte, koja omogućava prepodešavanje unutar samog programa.

Daleko najvažniji elemenat cele priče je poslojanje ili neposlojanje veznika za *AutoCAD V 11*, odnosno tzv. „enhanced 386" režim. „Trident" ga ima, „Tseng" ga nema. Najblaže rečeno, ovo je veoma čudno od firme „Tseng", sve ko ko lole pretenduje na ozbiljnosi morao bi da ima ovaj veznik, jer se radi o apsolutnom standardu za ovu vrstu programe. Nemati ga, lo je isto kao kada neki program za obradu teksta ne podržave „Epson" i HP štampače – ka tastrole!

Naravno oba proizvoda imaju veznike za sada već obavezni *MS Windows 3.0* (a i za slanje verzije istog programa), kao i za „Lolus" proizvode.

NA BRZINSKOM TESTU...

Priložene slike kazuju svoj deo priče. Slika 1 prikazuje rezultate rada koje su kartice izvele na dalim tekstovima u tekstualnom rezimu, a slika 2 rezultate rada u grafičkom režimu u oba slučaja, radi se o hardverskom pogledu na stvari. Slika 3 prikazuje brzinu rada pod programom *MS Windows* – koji je valjda najrasprostranjeniji „težak" program (u grafickom smislu) u opticaju.

Kao platformu upotrebili smo našu R/Optimu 486 kako bismo u celini alminisali uticaj samog računara koliko god je to više moguce. Sva merenja su obavljena sa „glol" mašinom, odnosno baz ikakvih posebnih programa (kao recimo *QEMM* ili *386 to-the-Max*), kako bismo izbegli bilo kakve neželjene interakcije i dopunski sloj programa koji bi mogao zamutiti rezultate, all se uključenom funkcijom duplikatna memorije (*video shadow RAM*) koja je hardverska funkcija računara (pa prema lome, treba da bude trajno osposobljana).

Rezultati sasvim nadvosmisleno pokazuju da je „Tseng" kartica brža od „Trident" kartice. U kontekstlu brzine izbora nema.

Posebno nas je razočerala brzina „Trident" čipa na testovime pisanje, memorija-memorija i ekran-memorija. Tu je ova kartica oselno podbacila u odnosu na „Tseng" karticu, a to su prilično važni elemenli baš za rad sa CAD programime (mada ce kod njih uvek najveće usko grlo bili sami proračuni elemenata slike).

U slučaju rada sa programom *Windows*, razlike su se osetno smanjile, iako se i dalje ostale u korist „Tseng" kartice. Ipak, u ovom slucaju su se svele na menje-više akademske razlike, jer objektivno nekoliko sekundi gore-dole ne znači baš mnogo.

...I U SVAKODNEVNOM ŽIVOTU

Obe kartice smo konstili i na dnevnoj bazi za svakodnevni rad po deselak dana. Moramo reći da neke bitnije razlike u brzini u svakodnevnom poslu nećete osetiti, tako one objektivno postoje. U praktičnom smislu, možete smatrati da su obe kartice jednako brze.

Tokom upotrebe, nismo naišli ni na jedan slučaj nepoželjnog ponašanja, meda napomnjemo da iz čisto praktičnih razloga nismo bili u mogućnosti da provanmo sve vezne programe za sve programske pakete. Ono što jesmo pogledali (*Windows* sve verzije, *WordPerfect* 5.0 i 5.1 i *Lotus*) radi onako kako treba.

Jedina blaga zamerka se odnosi na „Trident" karticu, odnosno njen početni ekranski font, koji ima pomalo čudno oblikovano malo slovo „v", no ta boljka se lako rešava brzim učitavanjem nekog drugog ekranskog fonta pomocu spoljnih programa, a ni sema po sebi nije kapitalna.

U osnovi, obe karte lepo rade posao kome su namenjene. „Trident" kartica je kao celovit proizvod, veoma mnogo popravile svoj opšti imidž, nekada je bilo problema se veznim programima, kako po kvalitetu, tako još više po broju, ali to je vreme očigledno definitivno prošlo. Sada je to moderan proizvod, sa izvanrednom izvedbom i pristupačnom cenom, solidnim priručnikom koji ipak traži doradu, čiji je veliki adut posedovanje pobude ze rad sa *AutoCAD*-om *V 11*.

"Tseng" kartica je neosporno tehnološki razvijenji proizvod, koji svoju cenu pravda brzinom rada, iako jeste od malo do dosta brža, ujedno je i relativno skupija. Priručnik je bolji od „Tridenl-a posebno u kontekstu programera, ali i uopšte. Sa druge strana, kvalitet veznika nije na nivou „Trident" kartice (kakva ironija! a nedavno je bilo upravo obrnuto!), i što je najgore, nema veznik za *AutoCAD V.11*, što je u današnje vreme ozbiljna mana (bar ako konstite ovaj program).

Ove kartice i kod nas prodaju mnogi trgovci, primera radi, „Tseng" kartice možete kupiti kod firmi „Gama" i „Most", a „Trident" karticu kod firmi „Intersoft", „Šutlić", „Genalec" i drugih.

HARDVERSKI KEŠ

O neophodnosti keširanja tvrdih diskova pisali smo već nekoliko puta, ali smo se pretežno zadržavali na softverskim rešenjima. Kako to izgleda kada se stvar uradi u hardveru?

Dejan V. Veselinović

Kada smo pre nekoliko brojeva „Računara" razmotrili razne programa koji se kod nas koriste za keširanje tvrdih diskova pokazali smo da je program *PC Kwik* dalako najbolje rešenje ne samo po brzini i efektima nego i po fleksibilnosti iako smo čuli nekoliko kritika na račun ovog programa nismo ih uzimali previše ozbiljno, pošlo sve one poriču isključivo od programere koji po definiciji nikad nisu zadovoljni Na svim platformama na kojima smo ga mi instalirali (ni jedne na pripada progremerima!) nismo sa njim doživeli ni najmanju neprijatnost

SNAGA SOFTVERA

Ovaj program sadrži niz modula za ubrazavanja rada tastature ekrana i diska, kao i mali đumemoriju za štampenje (printi spooler) i RAM disk ali njegov osnov čini program za keširenje diska Uzgrad iz ovog programa je izveden i program koji se nudi u okviru paketa *PC Tools*, all sa kudikamo manjim brojem opcija Iako *PC Kwik* sem uglavnom dobro odabera sve potrebne vrednosti pomoću priručnika ga možala dalja optimizovati, odnosno ubrzali mu rad za deljih desetak ill više procenata

Inharantna mana svih ovakvih programa pa semim tim i ovog je što sa jedna strana koriste sistemsku memoriju (čime se umenjuja raspoloživa količina što može biti veoma nepoželjno ako imate taman koliko vam treba za recimo CAD), a sa druge strane umanjuju već lonako mizernu količinu DOS memorija pa makar za desetak kilobajta (što po definiciji usporava računar u radu) S druge strane drastično ubrzavaju red sa diskom pa se sve gubici na drugim straname potiru i bogato nadoknađuju

Medutim dobro se zna da su ovakvi programi zapravo samo surogati zamene za „ono pravo" a to su inteligentni kontrolen tvrdih diskova Ovakvi uređaji su odavno poznati i odavno se koriste na velikim serverima a ono što ih je sprečavelo da predu u širu upotrebu je cene Ona nije mogla da padne dok proizvodnja ne postane masovna, a masovna proizvodnja nije moguće bez dobra cena – klasičan začarani krug

DC-600

Pa ipak Tajvanci (a ko bi drugi?) su izgleda nakako uspeli da razbiju taj krug i na tržišta pošalju čak nekoliko modela inteligentnih kaš kontrolera Oni su se, opet kod nas dosta brzo pojavili pa smo probali jedan modal

Probali smo ga uz naredna pretpostavka radi se o IDE kontroleru (dakle najpopularnijem formatu današnjice) koji je po definiciji interesantan za mnogo korisnike koji ja u nejdoslovnijem smislu reči inteligantan (ima svoj procesor i memoriju) i od koga očekujemo ugodnost i brzinu rada uz minimum problema a sve to pod raznim upravljačkim sistemima

I da „začnimo" – mora de bude razumna cena što će reći da mora biti jaftiniji od diska koji ubrzava Ovakvi uređaji ipak spadaju u profesionalni domen pa se pod tim uslovima može pretpostavil da će disk imati kapacitet od najmanje 180 MB

DC 600 je jedina oznaka na kaš kontroleru koji smo dobili na probu „DC" je valjde od „Disk Controllar", a ono „600" zelsta ne znamo čemu služi Kertica zahteva 16-bitnu utičnicu a ima dimenzije pune dužina mada je izvedena u nižem PC a na AT formatu Konati tehnologiju površinskog lema

Funkcionalno koristi procesor „Intel" 80186 na 12 MHz, ovo je hardverski integrisanija varzija dobrog, starog procesora 8086 koja objedinjuje sva periferijske čipova neophodna za rad ovog procesora Naše početno razočaranja ovim hibridom (16 bita iznutra 8 bita ka spoljnom svetu) donekle je stišano prednostima konfiguracija jer omogućava upotrebu jednog jedinog SIM modula umesto dva

Na kartici se nalaze četiri SIM postolja (sa metalnim držačima, hvala bogu) a u njih možete staviil module od 256 KB 1 MB ili 4 MB čime se dobija ukupno 16 MB potencijalna kaš memorija a to je već impresivna količine Čak možeta i malo mešali, malo jednih, malo drugih SIM-ova, mada je broj kombinacija ipak ograničen

Uz karticu nama bilo kakvih programskih disketa – sve je na njoj, u EPROM-ima Jedan sadrži opcije a drugi vi sami praprogramirate unošenjem važećih podataka Možete odrediti sva i svašta od parametara diska odnosno diskove preko opcija keširanja do upravljačkog sistema Nudi vam sa optimizacija za *DOS*, *Noveli UNIX* i *OS/2* – dakle sva Manji su izuzetno logično sređeni i jasni pa je podešava nja u najmenju ruku veoma lako i komforno Ovo smatramo velikim plusom

Druga veoma neobična osobina kartice je da ima odvojena priključke za dva IDE diska Svaki od njih podržava po dva diska i to tako da sabira kapacitela dva redno vezana diska E ovo je već izvanredno rešenje Izračunajte i sami – umesto kupovine velikih diskova, možela uzeti prvo naki manji pa naknadno dodati koliko vam trebe, a sve to tri puta! Ili možete odmah ili poslapeno kupovati IDE diskove od 570 MB pa povezivanjem na red imati funkcionalno dva diska od po 1 14 GB Do sada nismo vidali ništa ni nalik ovome sem u „Compaq" modelu „Systempro" a s obzirom na razliku u ceni jasna je prednost ova kartice

Uz sve to ostaja i ponaka dilema Pra svega na razumemo zašto su se projektanti opredelili za hibridni procesor, iznutra 16-bitni a spolja 8-bitni kada su za razliku u ceni od svega nekoliko dolara mogli da kupe pravi 16/16-bitni procesor 80186 na na 12 već na 16 MHz Time bi se performansa popravile za oko 35-55% uz malu razliku u ceni ili u krajnjoj liniji, zašto nije upotrebljen procesor 80286 pa ma kar i kao brzi 8086 koji se sada takođe može dobiti veoma jeftino a koji je još brži jer ja još inteqnsanij?

KONTROLER R AKCIJI

Za merenje performansi uzeli smo ovu karticu a kao protivnika poslavili smo joj program *PC Kwik* i to stanje izdanja verziju 3 51 (sada se prodaje verzija 4 1 a glavna razlika je u algoritmu sa keširanja diska)

Kao test poligon uzeli smo našu R/Optimu 486 rukovodeći se logikom da će ovakve ili još brža platforme biti glavni kendidati za servere Disk koji smo upotrebili za osnovna merenja je bio „Quantum" 170A a ceo sistem je imao 4 MB memorija

U prvoj koloni, prikazana su osobine diska bez ikakvog keširanja koristeći DOS 3 30 sva merenja su obavljena na particiji C koje je namamo bila 50% popunjena Svrha je da se što bolje simuliraju realni radni uslovi Naravno računer je bio korišćen bez ikakvih posabnih programa za optimizaciju (kao recimo *386-to-the-Max QEMM* ito)

Druga i treća kolona prikazuju performanse kada sa na gore opisanu konfiguraciju stavi samo program za kaširanje iz pakela *PC Kwik* i to sa opcionom komandom „EM" (minimalna upotreba DOS memorije) Druga kolone prikazuje kaširanje sa 1 MB a treća sa 3 MB sistemske memorija

Četvrta i peta kolona prikazuju razultata merenja sa DC-600 opet sa 1 i 3 MB

Razultati merenja su u najgorem slučaju veoma interesantni Merenja bez ičega, odnosno rezultate iz kolone jedan smo označili indeksom 1 00 U mašinskim lesiovima DC-600 je pokazala impresivne rezultate sa izuzelkom pristupa diksu kroz DOS u ovim lasiovima kartica je dala otprilike duplo gora rezultata Ovo je međutim sasvim normalno i očekivano jer pre nego što se pristupi podacima sa diske potrabno je pored običnog DOS-a proći i kroz sistem na samoj kartici Za uzvrat vreme prislupa disku je oko 46 puta (!) brža

U programskim laslovima stvar izgleda sasvim drugačija *PC Kwik* je mrlav-hladen potu kao keš karticu u sve tri grupe leslova Razlika od na izgled svega desetak procenata je resl no veća kada se ima u vidu de se indeksiranja baza podataka u sivarnosti radi dosta ralko odnosno najređe od svih pet testova le da je u tom leslu kartica dala sjajne rezultata koji su učestvovali u ukupnom proseku Kada se ovi razultati izuzmu razlika u brzini rasle na oko petnaestak procenata u konai *PC Kwik*-a

Drugi elemenai koji treba imati u vidu je opet sasvim normalan i očekivan kada obim lest datoteka prevaziđa raspoloživu memoriju karta neglo uspori Ono što je dosta neočekivano je koliko uspon u poređenju se *PC Kwik*-om čak i ze preko 50%

Ovde trebe imati u vidu da je brzina pristupa memoriji na keš kartici konstantne dok brzina keširanja sistamskom memorijom zavisi od brzine pristupa memoriji procesora. Drugim rečima navedene razlika važe za platforma sa procesorom 80486 na 25 MHz Isti procesor na 33 MHz će dati još bolja razultata se *PC Kwik*-om, ali recimo neki SX ne 25 MHz će zaostati

Tabela 1 Hardver ili softver

	Sam disk	1 MB	PC Kwik 3 MB	1 MB	Keš kon. 3 MB
Pristup kroz BIOS					
– slučajno	15,27	15,27	15,27	0,33	0,33
– redno	6 90	8 92	8 92	0 28	0 27
Pristup kroz DOS	19,11	19.50	21,26	38,83	35,98
Male datoteke					
512 × 512	50 69	5,38	2 96	3,62	3,79
64 × 4096	5,88	2 53	3 85	0 99	0 99
Indeks	1,00	7 15	8 30	12,27	11,83
Velike datoteke					
16 × 16384	3 40	0,28	0 27	0,88	0,82
8 × 32788	2,60	3 18	2,84	0,62	0,62
Indeks	1,00	1 73	2,06	3,53	3 66
Kumulativni indeks (mašinski test)	2,00	6 88	10 38	15 80	15 49
Prosečni indeks (mašinski test)	1,00	4,44	5 18	7,90	7 74
Varijabilne datoteke – male					
300 × 200	51,96	3,57	1,92	4,17	4 23
600 × 200	66,51	6,59	4 56	4 87	4 77
1200 × 200	81 53	10,85	8 13	6,25	6,10
2400 × 200	108,98	15,00	12,80	9 23	8,12
4800 × 200	161 32	21 15	17 35	27,35	15 10
9600 × 200	282,99	74.98	25 33	94 32	26 31
19200 × 200	512,56	104 29	71,34	164,12	110 12
Indeks	1,00	5 27	8,81	4 01	7 09
Varijabilne datoteke – velike					
30 × 2048	32,13	1 76	1,82	3 84	3,65
60 × 2048	36 58	2 80	2,19	4,07	4,12
120 × 2048	41 86	5,43	2,75	4 94	4 78
240 × 2048	46 06	11 42	8,47	8 05	5 99
480 × 2048	59 81	25 42	16 97	20 93	8 57
960 × 2048	81,18	60 49	34 43	92 71	16 69
1920 × 2048	124 46	90.06	69,59	144 51	100 67
Indeks	1,00	2 08	3,11	1 53	2 93
Unitest baza podataka					
Pretraživenje	146 18	54 80	81,18	63 86	62,95
Računanje	117 05	64 59	49 63	59 45	56 32
Indeksiranje	568 65	174.99	166,99	63.03	56 20
Indeksno pretraživenje	94 41	11.77	11 28	17 45	16,99
Sortiranje	61 64	74 88	41 56	72,77	52 09
Indeks	1,00	2 56	3 06	3 55	3,98
Kumulativni indeks (programi)	3 00	9,93	14 96	9 09	14,00
Prosečan indeks (programi)	1,00	3,31	4 99	3,03	4,67

NAPOMENA Merenja pristupa disku kroz BIOS i kroz DOS se ne boduju navedene su isključivo radi pokazivanja uticaja raznih rešenja na ova dva faktora

Performanse keš kontrolera IDE tvrdih diskova (opis u tekstu)

iza ove kartice Dakle dok su efekti *PC Kwik*-a vezani za sistemsku memoriju kartica nije pa su njene dobiti na sporijim mašinama utoliko veće ukoliko je računar sporiji

KAKO ZA KOGA

Dati konačnu ocenu o ovoj kartici zaista nije lako Zato pođimo redom

Za korisnika-pojedinca ova karta nema mnogo smisla Ono što biste dali za nju bolje dajte na memoriju koju ćete upotrebiti kao keš memoriju pomoću programa *PC Kwik* U neće-lu za cenu karte možete kupiti oko 5 MB memorije koje po potrebi može biti i nešto drugo, a ne samo keš memorija

Za servere srednje veličine i namene ova karta može biti dobar posao ali i ne mora Ako nemate nameru da vezujete više od dva diska u IDE tehnologiji (a sada ih ima i od 570 MB) verovatno vam neće trebati karta i verovatno će vam sasvim lepo posao obaviti još malo sistemske memorije odvojene za keširanje Ovo se posebno odnosi na mreže pod *Novell*-om, koji ima sopstveni potprogram za keširanje za koji se zna da je odličan

Za servere namenjene većim mrežama sa jačim zahtevima ova kartica bi mogla obaviti veoma koristan posao a u nekim slučajevima mogla bi biti čak i od kritične važnosti Činjenica da podržava faktički četiri diska je veoma značajna a podatak da se može optimizovati za razne namene takođe govori u prilog njene upotrebe Međutim ako vam u sistemu treba sva sistemska memorija a sistem je povezan u mrežu onda ova kartica postaje apsolutna nužnost jer sa 16 MB sopstvene memorije može mnogo da učini za performanse sistema

Sve u svemu interesantan proizvod koji ipak ostavlja neke sumnje u koncepciju a pre svega upotrebu hibridnog procesora umesto čistokrvnog 16/16-bitnog čipa na još većem taktu Može biti veoma vredan ali pre kupovine ipak dobro razmislite a još bolje se preračunajte Jer cena kartice sa 16 MB je otprilike onolika koliko košta i prosečna SX mašina na tržištu, a to nije malo

Prenosni računari

COMPAQ 486C COLOR



Dok se proizvođači trude da po svaku cenu što više smanje prenosne računare Compaq jedan od lidera na tržištu prenosnivih odnosno prevozivih računara, krenuo je sasvim drugim putem

Nakon izuzetno uspešnog modela SLT Compaq se opet posvetio posebnoj grupi korisnika novim modelom *Portable 486c Color* Radi se o prevozivoj kutiji (*lunchbox*) koja uopšte nema baterije čime se jasno stavlja do znanja da nije namenjen brzim putovanjima Umesto toga ovaj računar nudi procesor Intel 80486DX na 33 MHz sa drugostepenom keš memorijom Sabirnica koristi EISA tehnologiju što znači da je računar i naknadno proširljiv

Nude se dve veličine tvrdih diskova model 120 ima tvrdi disk od 120 MB a model 210 disk od 210 MB Oba sistema imaju standardno ugrađenu disketnu jedinicu od 3 5 inča kapaciteta od 1 44 MB i standardno se isporučuju sa po 4 MB sistemske memorije

Međutim udarni aspekt ovog računara je njegov prikaz koji koristi aktivnu TXT tehnologiju u punom spektru boja VGA standarda Dosadašnji prikazi su koristili multipleksiranje odnosno jedan tranzistor je naizmenično menjao stanje više piksela u aktivnoj tehnologiji svaki piksel ima sopstveni tranzistor za određivanje opšteg stanja (On/Off) i drugi tranzistor za određivanje boje Prednosti aktivne tehnologije su u veoma osetno bržem menjanju stanja boljoj definisanosti prikaza i jasnije definisanim bojama Problem je naravno cena koja je osetno veća no u slučaju pasivne tehnologije

"Većina prenosivih računara sa prikazima u boji iako su najavljeni ove godine neće se pojaviti u prodaji do kraja godine Compaq je uvek bio predvodnik sa novim tehnologijama a trenutno se bavimo tehnologijom kolor prikaza za prenosive mašine Planiramo da budemo veoma agresivan učesnik u ovoj oblasti" izjavio je Ekhard Pfajfer (Eckhard Pfeiffer) predsednik firme

Zainteresovani za ovakve modele veoma moćnih prenosivih mašina su profesionalci iz raznih oblasti koji koriste veoma složene programe – arhitekti inženjeri i programeri u oblasti CAD/CAM-a finansijski stručnjaci ljudi kojima trebaju izuzetne mogućnosti prezentacije i drugi *(D V)*

Korisna adresa Compaq Computer GmbH Elektrastr 6 8000 Muenchen 81 SR Nemačka

ZA SVAKI JEZIK NA SVETU

Devedeset šest znakova, koliko podržava ASCII standard, dovoljno je taman da se predstavi engleska azbuka, a ko ne koristi engleski – mrzi ASCII set. Dva nova standarda pretenduju da reše problem.

Vladimir Kostić

Problem sa ASCII standardom je to što je nastao u Americi. Amerikanci imaju tendenciju da sebe smatraju za centar sveta, pa je tako u ASCII set ušlo 26 velikih i 26 malih slova engleske abecede, 10 brojeva, i nešto interpunkcijskih znakova. Raznorazna slova koja se koriste u ostatku sveta su jednostavno i elegantno ignorisane, što je omogućilo da ASCII set stane u samo 7 bita (osmi se često koristi za proveru parnosti u komunikacijama). Velikir IBM je ASCII set dopunio sa još 128 znakova, ali to su uglavnom razni grafički karakteri. Samo je nekoliko „velikih" evropskih jezika kao što su francuski ili nemački našlo podršku u okviru tih dodatnih 128 karaktera. Mada je široko rasprostranjen IBM-ov 8-bitni karakter set ipak ne predstavlja standard.

KOLIKO NIJE PREVIŠE?

Korisnici koji ne govore engleski osuđeni su da krpe ASCII set i dodaju slove koje nedostaju. Francuzu nedostaju akcenti, Šveđaninu cela slova. Rusi ne mogu ni svoje ime da napišu kako treba. Vlasnici kompjutera kod nas to dobro znaju – ostali smo uskraćeni za 10 slova (ne računajući celu ćirilicu).

Da bi sa rešio problem napravljen je YUASCII set koji važi za standard ali žrtvuje nekoliko važnih specijalnih znakova. U želji da se to popravi, nastalo je još nekoliko nazovi standarda kao što su LATIN2 „Računari" „Moj Mikro" itd. Najzad dobar deo korisnika se jednostavno odlučio da ignoriše naša slova i živi sa nepromenjenim ASCII setom. A da se ASCII setom ipak može preživeti najbolje će posvedočiti urednik „Računara" kome je ovaj tekst prosleđen bez kvačica i kukica.

Ipak nešto se moralo uraditi.

Ideja da se ASCII set proširi sa još nekoliko bitova nije nova. Ustvari već godinama se govori o tzv. „super alfabetu" i nekoliko su dva predloga. Prvi koji je nazvan *Unicode* i koristi 16 bita je sponzoriran konzorcijum američkih hardverskih i softverskih firmi a drugi koji koristi čak 32 bita je predložila Međunarodna organizacija za standarde pod nazivom ISO 10646. Sa 16 bita se može kodirati 65536 karaktera a sa 32 čak 4 milijarde. Prosečnom čoveku se od ovih cifri vrti u glavi, naročito ako mu je hard disk sasvim dobro popunjen običnim 8-bitnim karakterima. Ipak u današnjem svetu postoji puno jezika i dijalekata a mnogi konsole sopstvena pisma. Kada se to sve sabere treba puno bitova, samo koliko?

NI 16 NIJE DOVOLJNO

Tvorci *Unicode*-a su sabrali važnije jezike sveta i zaključili da bi nekih 12 ili 13 bita moglo da posluži. All ne leži vraže – problem su jezici koji uopšte nemaju glasovno pismo nego koriste slikovno pismo koje često sadrži na desetine hiljada ideograma. Da primer, kineski, korejanski i japanski. Da stvar bude još gora, sve su to velike nacije sa puno stanovnika i puno kompjutera, pa se ne mogu jednostavno ignorisati – kao što bi to bilo moguće sa nekim plemenom usred Amazonske džungle. Tvorci *Unicode*-a su pošli od činjenice da la tri jezika odnosno njihova pisma vuku korene od zajedničkog pretka – starokineskog. Eliminisali su sve što i malo liči jedno na drugo, sve lepo sabili u set od 45 000 znakova i stvar nazvali *UniHan*.

E tu su počela da se lome koplja. Lingvisti vrište kako su ta tri jezika međusobno taman toliko slična koliko i engleski, ruski i španski. Da ne pričamo o povređenim nacionalnim osećanjima. U odbranu *Unicode*-a stoji činjenica da je sve lepo stalo u 16 bita. Takođe prvih 128 karaktera *Unicode*-a su po ASCII standardu što garantuje koliko toliko lak prelaz na novi standard.

Tvorci ISO 10646 predloga naprotiv nisu bili željni kompromisa. Svaki jezik sveta je trebalo da uđe u karakter set i ne samo to, nego je sveko pismo svakog jezika trebalo da zadrži svoj integritet. Drugim rečima ISO 10646 reda sva pisma sveta jedno za drugim, pa ponavlja se neka slova ili ne. Koliko je to važno znaju programeri koji su se mučili sa YU sortiranjem. Pošto 32 bita omogućavaju kombinih 4 294 967 296 kombinacija, ušli su i kompletan kineski, korejanski i japanski.

Eh, da, ali zar 32 bita po karakteru nije malo previše? Uostalom, kome konkretno trebaju svi ti silni karakteri? Dobro, nekome možda ipak trebaju, ali zar je to razlog da sve na hard disku i u memoriji zauzima četvorostruko veći prostor?

RAT DVA STANDARDA

Dva nova standarda naravno znače buduću nekompatibilnost, naročito ako je jedan 16-bitni, a drugi 32-bitni. Srećom ni jedan od ta dva predloga trenutno nije zvaničan standard, pa su učinjeni neki koraci da se oni ujedine. Konkretno predlog je da *Unicode* bude podskup ISO 10646 standarda. To, naravno, nije teško ostvariti – ISO 10464 je dovoljno veliki da bi mogao da sadrži 65536 raznih *Unicode*-ova. Time bi automatski i ASCII postao podskup ISO 10464 standarda, pošto je već podskup *Unicode* standarda.

U stvari ISO 10464 bi mogao da se koristi na tri načina:

- Kao 7 bitni podskup kompatibilan sa ASCII standardom i svim živim od opreme i softvera što danas postoji.
- Kao 16-bitni podskup kompatibilan se *Unicode*-om. Za većinu aplikacija *UniHan* bi bio sasvim dovoljan, a veličina od 16 bita bila sasvim prihvatljiva.
- Kao kompletan 32-bitni set sa svim živim pismima sveta.

Većina posmatrača veruje da bi 16 bitni *Unicode* podskup postao *de facto* industrijski standard, a da bi se puni 32 bitni set koristio samo u specijalnim okolnostima, ogiavnom na tržištima Kine, Koreje i Japana.

Ideja *UniHan*-e je doživela i važan blagoslov Japanskog instituta za standarde pod uslovom da i kompletno japansko Kanji pismo bude deo standarda, što je uslov koji ISO 10646 ispunjava.

ocedurama nižeg nivoa.

ISO 10646 bi trebalo da bude proglašen za zvanični međunarodni standard tokom ove godine. Sem ako ne bude iznenađenja, a već ih je bilo. ISO 10646 je već jednom pao na glasanju za zvanični standard, jer se nekima ideja *Unicode*-a nije dopala. U međuvremenu je stvar malo određena, ali lingvisti (naročito oni iz Japana, Kine i Koreje) i dalje vrište protiv *UniHan*-a i nisu jedini koji su u ISO 10464 videli mane i nedostatke. Postoji li uopšte na ovoj planeti standard koji sve zadovoljava?

KORAK DALJE

Korak dalje je napravio neizbežni „Apple" (a ko bi drugi?). Polazeći od *Unicode* standarda „Apple" je predložio formal zapisa teksta koji bi sadržao i informacije kao što su font, stil, veličina pa čak i izgled paragrafa. Stvar se zove *Rich Text* i nije daleko od pameti ako se zna da se jedan jezik piše sleva nadesno, drugi zdesna nalevo a treći odozgo na dole.

Rich Text bi takođe bio Božiji blagoslov kada treba prenositi podatke iz jedne aplikacije u drugu. Ko je ikada pokušao da prenese podatke iz jednog teksi procesora u drugi teksi procesor na drugoj mašini i pod drugim operativnim sistemom znaće da ceni prednosti *Rich Text*-a.

Kada jednom ISO 10646 bude proglašen za zvanični standard, treba ga primeniti – ili bar pokušati. Možda bi bilo bolje da umesto „Keko ga primenili?" pitanje glasi „Da li se ISO 10646 uopšte može primeniti?"

Jednom tekst procesoru, na primer, može biti svejedno da li su karakteri dugački 8 ili 32 bita. Naravno, ako izuzmemo to da će raditi mnogo sporije sa 32 bita i da treba rešiti problem prikazivanja takvih znakova na ekranu, problem njihovog štampanja i problem tastatura. Šta vredi ako imate 45 000 *UniHan* karaktera ako nemate r tolko tastera? Jeste li ikada videli japansku pisaću mašinu? Sačuvaj me bože te naprave!

Napraviti 16-bitni ili 32-bitni super-alfabet na papiru nije neki problem i samo je prvi korak. Zamislite video kartu se 32-bitnim karakter generatorom ili tek printer koji u ROM-u sadrži definiciju za 400 000 slova u 10 fontova, svaki u po 20 veličina!

Ne, ASCII će još dugo živeti – u to nema sumnje!

ŽIVOT NA SABIRNICI

Na današnjem tržištu mikroračunara vlada veliko šarenilo raznovrsnih interfejs standarda za povezivanje računara sa spoljnim memorijama, periferijama, kao i drugim računarima. Da stvari budu gore, ti standardi su, po pravilu, međusobno nekompatibilni. Najzad, pre nekoliko godina na tržištu se suvereno ustoličilo optimalno rešenje za ove probleme – SCSI standard.

Nebojša Novaković

Na tržištu postoji čitava gomila standarda interfejse. Za serijska veza definisani su RS232, RS422 RS423 za paralelne veze preovlađuju „Centronics" i IEEE488 (GPIB) za mreže „Ethernet", „Arcnet" „Token Ring" FDDI HIPPI (ovaj poslednji brzine 1 Gbit/s kod „Cray" superračunara) a za diskove ST506 IDE ESDI SMD IPI-2. Na ovu gomilu standarda treba dodati i mnoštvo specifičnih interfejse na pojedinim računarima koje podržava samo njihov proizvođač (setimo se prestarog C64 ili „atari ST"). Bilo je više pokušaja da se takvoj zbrci učini kraj sve dok se nije pojavio SCSI standard o čijoj novoj verziji SCSI-2 govorimo u ovom tekstu.

MALA SCSI ISTORIJA

Početkom prethodne decenije pojavila se potreba za brzim i fleksibilnim paralelnim interfejsom namenjenim prevashodno vezi sa spoljnim memorijama. Sabirnica (bus) interfejsa trebalo je da ima baram 8-bitnu širinu i da podržava brze prenose blokova podataka između računara i spoljnih jedinica (tvrdog diska) brzinom većom od fizičke brzine same spoljne jedinice. Ta veća brzina postizala bi se upotrebom međumemorija i brzim kontrolerima. Prvi takvi standardi bili su IBM OEM Channel za prastare grdosije serije 360 i kasniji ANSI IPI (Intelligent Peripheral Interface) koji je zajedno sa svojim bržim naslednikom IPI-2 takođe ostao u sferi malo većih kutija.

Nekako u to vreme početkom osamdesetih kompanija „Shugart" poznati (bivši) proizvođač disketnih jedinica i diskova razvija svoj nezavisan standard paralelnog interfejsa za vezu sa spoljnim memorijama i periferijama i naziva ga SASI (Shugart Associates System Interface). Ovaj standard ubrzo zatim prihvataju i „Data Technology Corporation" (DTC) „Xebec" i „Western Digital". Kao što znamo „Western Digital" i DTC još i danes prave izuzetno dobre disk kontrolere svih standarda. SASI je ujedno bio i prvi mikroračunarski disk interfejs čije je praktične brzina prenosa premašila jedan megabajt u sekundi. Nekoliko godina nakon nestanka SASI-a, stvaraoci ovog standarda se povezuju sa timom inženjera koja je za ANSI razvijela ISI (Intelligent System Interface – nesuđenog naslednika IPI) i zajednički delimišu polazne osnove novog SCSI (Small Computer System Interface) standarda. SCSI, znači, nastaje kao univerzalni I/O interfejs za mikroračunare, namenjen na prvom mestu povezivanju sa spoljnim memorijama, ali i sa drugim periferijama koje zahtevaju brz pranos podataka. Prva usvojena specifikacija SCSI pojavila se 1986 godine, i odmah se kranulo sa dopunama.

OD SCSI-1...

Prvi SCSI je definisan kao dvosmerna 8-bitna paralelna sabirnica koja omogućava sekvencijalno vezivanje do osam uređaja, gde je broj 1 sam računar koji je ujedno i „bus master" odnosno domaćin – „host" vezanim periferijama. Usled toga ćete za SCSI kertice ne tržištu veoma retko naći naziv „SCSI controller", koji je u biti i nepravilan nego „SCSI host adaptar". U opisu SCSI 2 videćemo da ovaj limit od sedam uređaja na host adapteru uopšte nija tako strašan, odnosno da i bazične arhitektura SCSI omogućava praktično bezgranično proširivanje novim jedinicama.

SCSI ima jednu bitnu dodirnu tačku sa IDE (Integrated Drive Electronics) AT bus hard disk interfejsom, koja je u isto vreme ključna razlika u odnosu na ESDI i ST506 standard. Naime na SCSI kerfici nema hard disk ili drugih kontrolera (izuzetak su samo flopiji koji lonako nemaju veze sa SCSI) već samo „host adapter" kolo čiji su izlazi na jednoj strani sabirnica za vezu sa CPU a na drugoj sam SCSI. Svaki uređaj na SCSI sabirnici bilo da je hard disk optički disk, strimer, laserski štampač, skener ili nešto drugo u sebi onda ima ugrađen kontroler za vezu sa SCSI.

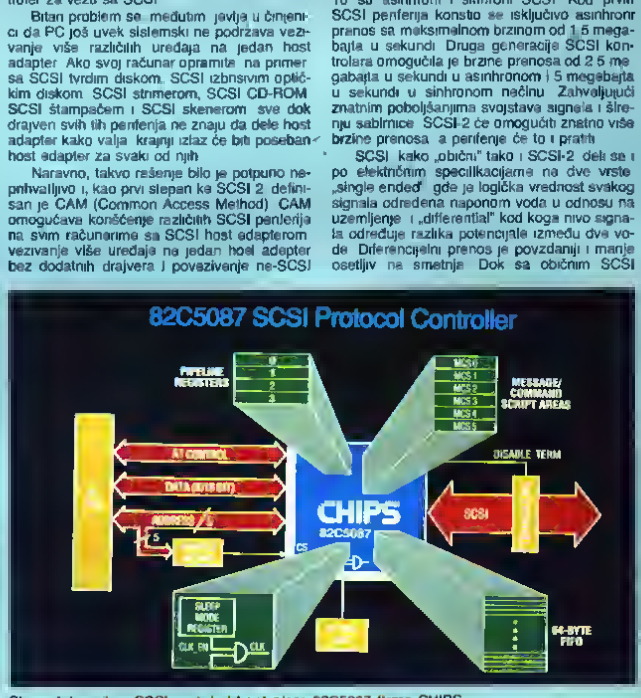
Bitan problem se, međutim, javlja u činjenici da PC još uvek sistemski ne podržava vezivanje više različitih uređaja na jedan host adapter. Ako svoj računar opremite na primer sa SCSI tvrdim diskom, SCSI izbrisivim optičkim diskom, SCSI strimerom, SCSI CD-ROM, SCSI štampačem i SCSI skenerom sve dok drajveri svih tih periferija ne znaju da dele host adapter kako valja, krajnji izlaz će biti poseban host adapter za svaki od njih.

Naravno, takvo rešenje bilo je potpuno neprihvatljivo i, kao prvi stepen ka SCSI 2 definisan je CAM (Common Access Method) CAM omogućava konsolidovanje različitih SCSI periferija na svim računarima sa SCSI host adapterom vezivanje više uređaja na jedan host adapter bez dodatnih drajvera i povezivanje ne-SCSI uređaja (na primer ESDI disk ili Ethernet interfejs) na SCSI bus preko konvertera. Istovremeno sa CAM definisan je i ATA (AT Bus Attachmani) koji određuje način na koji će SCSI kontroleri na većim hard diskovima oponašati stari IBM AT ST506 disk interfejs, to jest uklapati se u dobro nam znana BIOS ograničenja od 16 glava i 1024 cilindra. ATA je uz novu EATA (Extended ATA) specifikaciju temelj na osnovu kojeg su „Compaq" i „Western Digital" osmislili danas široko rasprostranjeni IDE standard.

Od 1987 nadalje popularnost SCSI linearno raste. Sve više proizvođača hard diskova svoje slitne 5 25" modela pored tada nepnkosnovenog ESDI i snira i u SCSI verzijama. Međutim krajem 1988 pojavljuju se prvi diskovi (Maxtor 8760) čija fizička brzina prenosa dosliže ili nadmašuje tadašnju brzinu SCSI od 1 5 megabajta u sekundi. Otkud sad to?

Naime SCSI-1 imao je dve podvrste koje se razlikuju po načinu odnosno brzini prenosa. To su asinhroni i sinhroni SCSI. Kod prvih SCSI periferija konsto se isključivo asinhroni pranos sa maksimalnom brzinom od 1,5 megabajta u sekundi. Druga generacija SCSI kontrolara omogućila je brzine prenosa od 2 5 megabajta u sekundi u asinhronom i 5 megabajta u sekundi u sinhronom načinu. Zahvaljujući znatnim poboljšanjima svojstava signala i širenju sabirnice SCSI-2 će omogućiti znatno više brzine prenosa a periferije će to i pratiti.

SCSI kako „obični" tako i SCSI-2 deli se i po električnim specifikacijama na dve vrste „single ended" gde je logička vrednost svakog signala određena naponom voda u odnosu na uzemljenje i „differential" kod koga nivo signala određuje razlika potencijale između dva voda. Diferencijalni prenos je povzdaniji i manje osetljiv na smetnje. Dok sa običnim SCSI



Shematski prikaz SCSI protokol kontrolera 82C5087 firma CHIPS

ukupna dužina kabla koji vezuje uređaje na SCSI host adaptar može biti najviše 6 metara kod diferencijalnog SCSI maksimalna dužina je celih 25 metara S druga strane diferencijalni SCSI je skuplji za implementaciju i koristi ga samo mali broj najbržih hard diskova visoke klase Bez obzira na mere sigurnosti preduzete da na bi došlo do kvara u slučaju priključenja „single ended" SCSI uređaja na diferencijalnu SCSI sabirnicu, bolje ga ne mešati jedno s drugim

Sve u svemu, zahtevi za većim performansama i „sužívotu" različitih SCSI uređaja na istoj sabirnici su uz neka dodatna unapređenja, vodili ka definisanju SCSI-2 lokom 1986 GODINE

...DO SCSI-2

SCSI-2 se kao manje-više potpuni standard pojavio 1989 Pogledajmo njegova osnovna elemanta SCSI-2 je, kao i njegov prethodnik paralelna sabirnica za vezu računara sa do sedam periferniih jedinice Osnovni SCSI-2 ima 8-bitnu širinu sabirnice kao i SCSI-1 a tim što nov „Fast SCSI-2" mod donosi praktičnu brzinu prenosa do 10 megabajta/s u sinhronom načinu prenosa Druga novost, „Wida SCSI-2" je proširanja sabirnice na 16 ili 32 bita preko posebnog konektora Brzi i široki SCSI-2 zajedno donose maksimalnu brzinu prenosa od celih 40 megabajta/s što je brza I od EISA sabirnice! Najbrži današnji SCSI-2 diskovi, kao „Seagate" Wren 9 ili „Elite" 3 implementiraju brzi i 16-bitni široki SCSI-2 tako da maksimalna brzina prenosa kod njih može da dostigne 20 megabajta/s prilikom prenosa iz bafera ka računaru

Konaktor SCSI-2 ima 50 pinova rasporedenih u dva reda po 25 „Single-anded" SCSI-2 koriati samo prvih 25 nožica dok diferancijalni SCSI-2 koristi svih 50 nožica Dužina kabiova su iste kao i kod starog SCSI Dodatni 68-pinski konaktor ja kod SCSI 2 namenjen proširanju na 16 odnosno 32 bita.

Kao što smo rakli SCSI-2 podrževa ukupno osam (računar i sedam periferija) jedinica na sabirnici pošto su tri bita na SCSI odvojena za identifikaciju jedinice Šta ako želimo da vežemo više uređaja? SCSI-2 nas i tada na ostavlja na cedilu jer svaka od sedam fizičkih jedinica može imati do osam logičkih jedinica (koja naravno mogu biti i fizički odvojena) a svaka od ovih logičkih jedinica može imati do 256 logičkih podjadinica! Bilo koji uređaj na SCSI. pe tako i ne SCSI-2 sabirnici može biti „initiator" koji izdaje naredbe i „target" koji ih izvršava Moguće kombinacija su jedan inicijator i jedan cilj jedan inicijator i više „ciljeva i više inicijatora i više ciljeva

U odnosu na SCSI-1 SCSI-2 je donao nekoliko novosti koje su omogućile znatno pouzdaniji rad viša uređaja na sabirnici preciznu standerdizaciju napajanja svih uređaja, paritatnu proveru podataka i arbitraciju pristupa različitih uređaja sabirnici sličnu arbitraciji procesorskih pristupa na EISA ili VME sabirnicama. Najznačajnija unapređanja, koja je u praksi omogućilo vezivanje potpuno različitih periferija na SCSI-2, je standardizacija obaveznog osnovnog skupa naredbi i poruka

SCSI-1, naime, nije imao precizno definisan skup naredbi za upravljanja periferijskim uradajima, s tim što je već bio začet osnovni skup obaveznih naredbi zajedničkih za sve uredaje — Common Command Set (CCS) SCSI-2 je taj skup osnovnih naredbi proširio i precizno fiksirao njihov format, kao i format dodatnih „davica-dependant" naredbi, tako da problama sa kompetibinošću sa ova strana više ne bi smalo biti

SCSI-2 naredbe sa od host adaptera ka uređaju šalju kao „Command Descriptor Blocks" (CDB). Struktura SCSI naredbi donekle podseća na naredbe savramanih CISC mikroprocesora. CDB tormati naredba su od 6, 10 i 12 bajtova. Kod svih formata prvi bajt je operacioni kod koji sadrži informacija o formatu i vrsti naredba i kod sama naredbe Drugi bajt sadrži broj uredaja koma se pristupa (upravo prva tri bita ovog bajta to definišu) i, zajedno sa trećim i četvtim bajtom logičku (na fizičku) adresu početnog bajta bloka podataka Zašto na stvarnu, fizičku adresu? SCSI sa ipak prevashodno koristi kao veza računara i spoljnih mamonja – diskova i traka. Svi znamo da sa datoteke koje se često menjaju i dopunjuju na tvrdom disku fragmantiraju na više različitih područja diska koja imaju različita fizička adrese (glava, cilindar, sektor) ali sa stanovišta računara logička adresa tih fragmanata programa nadovezuju sa Pretvaranja logičkih u fizičke adresa diska, posao sličan onom koji obavljaju jedinice za upravljanja mamonjom (Memory Managament Unit – MMU) u savremanim 32-bitnim procesorima kod stanh ST506 i ESDI standarda izvršava disk kontroler kartica u računaru SCSI host adapter s tim poslom nama ništa – sve obavlja kontroler koji je ugraden u samom disku Sličan posao obavljaju i kontrolen ugrađeni u IDE diskove Pati bajt sadrži informaciju o dužini prenosa podataka ili dodatne parametre potrebne da se naredba izvrši dok je poslednji bajt kontrolni 10, i 12-bajtni formati se jedino razlikuju po tome što dodatni bajtovi omogućuju duže adrese (do 2^{32} (4 Gb) ili 2^{48} (1 Tb)) ili veće blokova prenosa (do 2^{24} (16 Mb)) Po izvršenju naredbe računar nazad dobija statusni bajt sa informacijom o atanju sabirnice nakon izvršenja naredbe

Naredbe SCSt-2 iz CCS vezana su prevashodno za tvrda diskova i druge spoljna memorija COMMON COMMAND SET tako sadrži naredbe ze formatiranje jedinica prenos jednog ili više blokova podatake, čitanje iz ili upis na jedincu selekciju načina rad jedinice verifikaciju podataka Iražanja za odredanim podatkom ili blokom podataka upravljanja baterom u jedinici kao i proveru ispravnosti jedinice itd Pored ovih osnovnih naredbi svaki uređaj ima bar dasatek svojih dodatnih naredbi SCSt periferija koja nisu spoljna memorija, kao laserski štampač ploter, akaner, „Ethernet" adapter ili brzi modem (da i tol) namaju naks naredbe iz CCS koje imaju smisla samo kod spoljnih memorija ali a druge strana imaju veći skup svojih naredbi.

SCSI-2 podržava i „logging" – beležanja statističkih informacija i dogadaja na non-dala dalu diska. Ovi podaci o istoriji rada diska mogu se pozvati preko SCSI za analizu rada diska i mogu biti korisni u održavanju ili popravci aventualnog kvara. Na snažnijim SCSI (i svim SCSI-2) host adapterima podržano je i smeštanja komandi u red čekanja za izvršavanje u melom „queus" beferu Ne taj način, SCSI procesor može da nastavi primanje novih zahteva od CPU odk prethodna naredba još čeka u beferu na izvršavanje

Jedna od lepih osobina SCSt-2 je i lakoća međusobne komunikacije uređaja na SCSt-2 Ako na primer, vaš PC ima IDE ili ESDI hard disk i QIC strimer, arhiviranje sadržaja hard diska će prvo u velikoj meri angažovati procesor, drugo uzurpiraće sistemsku sabirnicu koja je možda potrebne grafičkoj karhci ili nekoma drugom Ako pak, imate SCSI-2 disk i SCSI-2 strimer ili DAT traku CPU će samo poslati naredbu, a ceo proces kopiranja više desetina ili stotina megabajta delje obavlja Inteligentni host adapter preko brza SCSI-2 sabirnice, a CPU može mimo da nastavi dalje svoj posao Isto važi, na primer. I za slanje slika smeštene na HD na laserski štampač oba ne SCSI To ujedno znači da tokom nekog rada u CAD na primer gde su i program i model ceh u RAM-u snažnog PC ili radne stanice bez ikakvog zamaranja procesora u pozadini možete obavljati poslova sa SCSI periferijama

SCSI-2 HOST ADAPTERI

Stari SCSI-1 i još više novi SCSI-2, izuzetno su fleksibilni interfejsi sa mnogobrojnim mogućnostima Kompleksni asinhroni i sinhroni protokoli prenosa podrška keširanju i međusobnoj komunikaciji perilerija, moraju biti implemanirani na način da što manje opterećuju sistem, a istovremeno zadovolje sve zahteve na polju performansi Sve to znači moramo imati brz i inteligentan SCSI host adapter koji će istovremeno biti lako ugradljiv u PC ili radnu stanicu, ako treba i na osnovnu ploču To, opet, znači da bi sve trebalo biti u jednom IC Pogledajmo razvoj SCSI host adapter čipova na primaru firma koja vodi na ovom polju, čuvenog NCR

Prve generacija SCSI čipova (NCR 5380) bila je neinteligentna i registarski orijentisana, kao i bilo koji kontrolar serijskih ili paralelnih portova Procesor je tu morao da obavlje praktično sav posao čak i oko samog SCSI protokola S druga strana konsnicima je odgovarala fleksibilnist takvih kola jer je drajver za tekav čip mogao da daje specifične informacije o bukvalno svakom stanju na SCSI sabirnici i na taj način lako testiranje SCSI pentenja Mogao se konstiti samo asinhroni protokol sa brzinom prenosa od 1 5 megadajta/s Prva generacija SCSI kola tražila je u proseku oko 4 000 linija koda za implementiranja drajvera.

Druga generacija (NCR 53C90 porodica), pored štedljivije i brže CMOS tehnologije, donosi i odredani nivo hardverskog izvođenja složenih sekvenci SCSI protokola, ubrzavajući tako rad Međutim, inteligancija nije bila bitno veća a fleksibilnost je bila manja jer su sekvence protokola bila fiksirana u hardveru, sa malim mogućnostima promana prako drajvera. Ovi kontrolen su već podržavali, pored brzeg asinhronog (do 2 5 Mb/s), i sinhroni način rada sa brzinom do 5 megabajta/a, kao i elemente ili ceo SCSI-2 Većina SCSI kola na sadašnjim SCSI i SCSI-2 host adapterima za personalce (WD. Adaptec Fujitsu) je iz ova generacije Druga generacija SCSI kola traži oko 2 500 linija koda za implementiranje drajvera

Tada se pojavila dilema u glavama dizajnera SCSI host adaptera Da li SCSI kola prve i druge generacija primenjivati sama ili u okruženju inteligentnih host adaptera?

Naintaligantan SCSI host u vidu jednog od gora opisanih kola zahteva odsta procesorskog rada ali ja jeftin i lako sa smašta na osnovnu ploču gde ga (ako to računar ima) pod svoje može prauzeti i naki I/O procesor opšta namane

Inteligantan SCSt host adapter je skuplji i mora biti na kartici Jedna bitna graška je prisutna kod gotovo svih inteligentnih SCSt kartica Imaju apore procesora (od 80186-10 od 80386SX-16) Ti proceson ipak nisu dovoljno snažni da vrlo brzo interpretiraju naredbe (bez dodavanja dodatnih milisakundi na prisljupno vrama hard diska, na primer) da na uzimamo u obzir vrama potrebno za međuprocesorsku komunikaciju preko AT, pa i EISA sabirnica A centralni procesori dostižu sve viša MIPS. I ta kašnjanja su za njih samim tim sve bolnija

Rašanja su SCSI host kola treća generacija To su specifični i vrlo brzi procesori koji upravljaju SCSI sabirnicom, minimizuju broj

prekida kojima bi „uznemiravali" CPU i obavijaju sve, pa i najsloženije zadatke bez ikakvog opterećenja centralnog procesora. Ove kola zahtevaju samo par stotina redova koda za drajver. Taj drajver treba samo da podržava posebne događaje (exceptions) SCSI kola i deja opseg adresa u RAM namenjen SCSI data baferu. Primar SCSI hosta kreće generacije je NCR 53C720 SCSI-2 Scripts procesor.

NCR 53C720 je najmlađi član 53C700 porodice SCSI hosi procesora treće generacija. 53C720 se neposredno vezuja na 33 MHz i486 procesor kao „bus master" sa sopstvenim DMA kontrolerom brzine prenosa do 106 megabajta/s na i486 strani ako se koriSta 2 1-1-1 burst ciklusi sa najbržom memorijom. 53C720 podržava brzi i 16-bitni široki SCSI-2. Tako na SCSI strani on dostiže brzinu prenosa do 20 megabajta/s po standardu, s tim što uz setovanje XFERP bits za tekt period translera na 0 la brzina može biti povećana na nešto većih ali i nestandardnih 25 megabajta/s

53C720 smešten u 208-pinskom SMT kućištu sastoji se od tri tesno povezane celine SCSI jezgra. DMA jezgra i SCRIPTs procesora. SCSI jezgro kao što smo rekli podržava 16-bitni brzi SCSI-2 kao i 8-bitni brzi i standardni SCSI-2. SCSI interfejs 53C720 je potpuno programabilan i omogućava dva nivoa rada. Prvi je „low level registar interface" koji kao i kod SCSI hostova prve generacija omogućava neposredan pristup CPU svim registrima i samoj SCSI sabirnici radi testiranja i dijagnostika. Povratna dijagnostika gde 53C720 sam sebi služi i kao inicijator i kao cilj takođe je veoma korsna u ia svrha. Drugi nivo je logički interfejs visokog nivoa, koji omogućava prihvatanje SCSI Scripts nareda iz RAM

DMA jezgro je bus master DMA koji se neposredno vezuja na sabirnicu „Intelovog" 486 na 33 ili 50 MHz a i na druga 32-bitna CPU uz malo dodatna logika. Sa svojim brzim 64-bajtnim FIFO baferom 53C720 podržava „burst" način prenosa na 486 sabirnici brzine do 106 megabajta/s sa mogućnošću memorija-memorija prenosa (ovaj DMA bi veoma lapo odmenio dvadasat puta sporije ugrađena DMA kontrolara u 486/AT čip setovima)

SCSI SCRIPTs procesor u 53C720 je brzi 2 MIPS (pazite ne dva VAX-MIPS-a ni RISC MIPS-a već dva miliona SCSI naredbi u sekundi!) procesor sa posebnim SCRIPTs mašinskim jezikom koji omogućava pisanja rutina za upravljanja SCSI i DMA jezgrima. kao i svim uređajima na SCSI-2 sabirnici. Ta rutine se izvršavaju neposredno uz RAM računara koma 53C720 pristupa jednako brzo kao i sam i486. Ono što je takođe veoma bitno ja da su SCSI Scripts rutina nezavisna od sistemskog CPU i sabirnice, to jasi ista rutina jednako će raditi na 53C720 bez obzira da li je on vezan na 486 AT 486 EISA, Macintosh III Sparc VME64 računar. SCRIPTs jezik omogućava lako i brzo oblikovanja rutina za upravljanja svim vrstama SCSI uređaja – od tvrdog diska do A0 termalnog plotera

NCR 53C700 porodica je veoma prihvaćena medu proizvođačima gratičkih radnih stanica. 53C720 na drugoj strani predstavlja veliki izazov za proizvođače najsnažnijih PC Razlog je jednostavan jedan jedini čip snažniji i brži od svih onih preskupih host adapter kartica a prilom višestruko javtinlji. omogućuje da se problem brze veze sa apoljnim memorijama i periferijama nova generacije elegentno reši Proizvođač je zaista ima u koja sa može pouzdati a procasoru svake čest. Uzgrad. paralalno se lansiranjem Intal 860XP, Motorola 88110, MIPS R4000 i „Texas Viking" SuperSparc procesora kao i završavanjem i568 (svi imaju jednu zajedničku odliku – 64-bitni data



SCSI obezbeđuja ogromnu brzinu pranosa i lati protokol za ave periferije

bus), NCR priprama novi 50 MHz SCSI-2 procesor sa neposrednom podrškom ultrabrzom selektivnom keširanju spoljnih memorija 64-bitnom sebirnicom brzina 320 megabajta/s na procesorskoj strani i punim 32-bitnim brzim SCSI-2 brzina do celih 40, odnosno 50 megabajta/s

PRIMENE SCSI-2

Zahvaljujući svojoj brzini. fleksibilnosti univerzalnosti i pouzdanosti (i da ne zaboravimo široko prihvaćenosti) SCSI-2 ima bezgranična mogućnosti primena Pogledajmo neka od njih pobliže

Brzi tvrdi diskovi

SCSi-2, za razliku od recimo ESDI, ne postavlja nikakve preprake da broj sektora u različitim zonama površine diska bude različit – unutrašnje zona fizički primaju manje, a spoljne zona više sektora. Podela diska na više (do 32) ovakvih zona zvana još i ZBR (Zoned Bit Recording), omogućava znatno povećanja ukupnog kapaciteta diska. kao i brzine prenosa u spoljnim zonama Drugo zbog naposredne podrška velikim baterima i inteligentnim keš-mamonjama na samom disku SCSI-2 omogućava dobro iskorišćenja svoja maksimalna brzina pranosa

Paraleini sistemi diskova

SCSI-2, pogotovo 32-bitna varzija, podržava „mirroring", ali uz pomoć dodatnih upravljača, i paralelne RAiD (Redundant Array of Inexpenelve Diska) sisteme Glavna odlika ovih sistema je što paralelnim povezivanjem više diskova daju znatno veću brzinu prenosa i daleko veću pouzdanost od jednog diska, uz izbegavanja dupliranja troškova običnog „mirroring" postupka Uzećemo jedan optimalan primer koji NCR naposredno podržava sa svojim novim RAID čip setom: Imamo ćelin Quantum ProDrive 1050 diska (3.5") 1050 Mb formatirano 10 ms prašupno vrama, 512 Kb keša 3 Mb/s prosečna lizička brzina prenosa s sa 8-bitnim Fasi SCSI-2 do 10 Mb/s) paralelno povazana, svaki sa svojim 8 bitnim SCSI-2 na RAID kontrolar

Podela je ustanovljena lako što ako u sistemu imamo 32-bitne reči svaki od ćelira diska nosi po jedan bajt. a peti periteini disk na dodatnom SCSI-2 konektoru RAID kontrolera nosi kontrolnu sumu sadržaja svih 32-bitnih rači. Brzi RAID kontroler na drugom kraju kao izlaz ima 32-bitni Fasi Wide SCSI-2 konektor kojim se vezuja na odgovarajući host adapter kao samo jedan od 8 uređaja na sabirnici Rezultujuće brzine ovakvog 4 Gb RAID sistema je fantastična 11 ms pristup (1 milisekunda dodata zbog vremena potrebnog RAID kontroleru i SCSI host adapteru za obradu blok-prenosa), 12 megabajta/s kontinualna i do 40 megabajta/s maksimalna PRAKTIČNA brzina prenosa sa polje diskova koje bi krajnjeg kupca koštalo oko USD 16 000 + dažbina Za smeštanje i reprodukcija sopstvanih filmova dušu dalo

Brzi laseraki štampači, skaneri i ploteri

Kod 600×600 dpi laserskih ili kolor termalnih sublimacionih štampača sa 16 miliona boja slike za štampanje zauzimaju nekoliko megabajta i plot fajlovi nisu zanamarljive veličine Umesto da se sve to šalje brzo sporih RS232 ili „Centronics" interfejsa, logično je iskonstiti brzinu i fleksibilnosti SCSI-2 i neće biti njiednog tajla (beram na u ovom tranuiku) čiji će prenos zahtevati više od sekundu-dve

Neposredno vezivanja sa drugim računarom

Ako imate potrebu da sa susednog računara prenesete racimo, 10 ili 110 Mb podataka, zašto bisle pravili nakakva nezgrapna bekapa? Povežite ih preko njihovih SCSI-2 host adaptera i prenos će biti gotov za najkraće moguće vrama!

Brzi merni i drugi naučni instrumenti

SCSI-2 je znatno brži i pouzdanlji od IEEE488, a nove generacije instrumenata često obrađuju velika količine podataka često u grafičkom obliku

SCSI'S THE WINNER

Na samo spoljna memorija nego i najveći deo mikroračunarskih periferija definitivno se odlučuja za SCSI Novi, brzi tvrdi diskovi ogromnih kapaciteta CD-ROM, sve vrsle optičkih diskova, strimeri, DAT sve je pod SCSi-2 kišobranom. Jedino još stan flopi nisu tu Ali nema vaze U dobu kada je 8 Mb minimum i za RAM i za veličinu bilo kog softverskog paketa tlopi od 1 2 1 44 pa i 2 88 Mb ne sma više da egzistira, sem oneko usput, za potrabe kompatibilnosti sa prošlošću Male 3 5" SCSI-2 magnetooptičke diskatica kapaciteta 128 Mb sama se nameću keo zamena Jedine ozbiljna konkurencija je IDE na polju manjih tvrdih diskova

Druga ganeracija SCSI standarda ispunila je sve zahteve postavljena davne 1986 u inicijalnoj specifikaciji SCSI Danas zaista imamo dobar i prihvaćen standard za vezivanja periferija na mikroračunara Šta dalje? Na SCSi-3 specifikaciji radi sa punom parom Zna sa samo to da će podržaveti automatsko konfigurisanje celog SCSI podsistema paralelni optički pranos i brzina reda 1 gigabajt u sekundi

SOFTVERSKA TRKA VEKA

Sjajan uspeh na tržištu najnovije verzije DOS-a nikoga ne zavarava – radi se o cedenju poslednjih atoma života iz jednog operativnog sistema koji je daleko zaostao za razvojem same mašinske osnove koju koristi. Njegova radikalna revizija se više ne može odlagati, a dojučerašnji tesni saradnici, IBM i Microsoft, ulaze u ovu utakmicu kao najljući protivnici – avako sa svojim rešenjem.

Dejen V. Veselinović

Kada je u julu 1981 godine tada malo poznata firma „Microsoft" kupile od još manje poznate firme „Seattle Computer Product" sva prava na upravljački sistem poznat pod imenom 86-DOS, niko nije obraćao pažnju. Međutim već u avgustu iste godine IBM je prikazao svoje viđenje ličnog računara, koji je konsti preređenu verziju ovog upravljačkog sistema nazvanu *Microsoft DOS*

Ta kupovina je bila valjda najznačajniji korak koji je „Microsoft" ikada napravio jer mu je omogućila da postane maltene monopolista na upravljački sistem namenjen računaru kome je bilo suđeno da postane apsolutni bestseler sveta u svojoj oblasti. Niko nikada ranije, a verovatno ni kasnije (?) neće imati oko 70 miliona (!) platformi sa svojim upravljačkim sistemom



EVOLUCIJA DOS-A

Evolucija DOS-a prikazana je na slikama 1 i 2. Slika 1 prikazuje nove verzije kako su se pojavljivale, a slika 2 rast DOS-a po količini memorije koju je zahtevao

Proizlazi da DOS 5.0 zauzima isto onoliko mamorije koliko i DOS 1.1 kada se ono što može pomeri u gornje blokove (to jest u HMA deo memorije iznad 1 MB). Nema sumnje da je ovo dobro, ali je isto tako jasno da se radi o krpežu, o cedenju poslednjih atoma života iz jednog sistema koji je daleko zaostao za razvojem same mašinske osnove koju koristi

Neće vam mnogo pomoći ni razne pošlapalice koje su se u međuvremenu pojavile kao što su razni programi za upravljanje memorijom 0 je *QEMM* ili *386-to-the-Max* pa čak ni ideja sa proširenom (*extended* ili LIM) memorijom pomoći hoće ali stvarno rašiti problem neće. Barijera od 640 KB i dalje ostaje glavna prepreka bržem i boljem računarstvu. Zahvaljujući njoj ma kakvu platformu imali, vi, u osnovi imate veoma brzi XT

Ako vam sve ovo deluje bizarno, napravite jednu malu probu. Pretpostavljajući da imate bar 2 MB sistemske memorije i procesor na bazi 32 bitne tehnologije (može i SX) pokreni te *Windows 3.0* u realnom i 386 režimu pa samo gledajte razliku u brzini i divite se

Nije nikakva tajna da se novi upravljački sistem na sav glas traži još od 1984. godine kada je IBM prikazao svoj model AT, se pravim 16-bitnim procesorom. Od onda naovamo, svi žele novi sistem, koji će evolucijom ispraviti sve nužne kompromise sadržane u DOS-u. Ali sve do danas to je ostao samo uzaludni vapaj

Zašto još uvek nemamo taj neki novi upravljački sistem, koji svi tako željno očekujemo? Iz ekonomskih razloga, zašto bi inače!

"TAMPON" PROGRAM

Shodno tradiciji, IBM i „Microsoft" su se prvobitno udružili oko razvoja tog novog upravljačkog sistema nazvanog OS/2 (*Operating System 2*), iako je ovaj savez lepo radio do pred kraj osamdesetih godina, u slučaju novog upravljačkog sistema veoma brzo je pukao – opet iz ekonomskih razloga

"Microsoft" je uporno od 1986. godine naovamo razvijao i čak prilično forsirao svoj tampon program *Windows*. Kažemo „tampon", jer je od početka i bio zamišljen kao posrednik između mašine i svega ostalog uključujući i nas kao korisnike. Tada se govorilo da je *Windows* zapravo pretvaranje PC-a u „Mac" pa će i „Apple" čak tužiti „Microsoft" zbog toga, zaboravilo se da je najveći lopov u svemu baš sam „Apple", jer je ceo sistem grafičke komunikacije smišljen u laboratorijama firme „Xerox" u Palo Altou još davne 1971. godine. Dakle, proizlazi da „Microsoft" nije smislio ama baš ništa novo pod kapom nebeskom, već da kasni nekih 15 godina

Međutim, *Windows* nikada nije uspeo da otkloni mane inherentne samom DOS-u a pre svega memorijsku barijeru. Čak ni novoj verziji 3.0/3.1 to nije pošlo za rukom, mada daleko bolje koristi raspoloživu memoriju od prethodnih verzija. A izvinjenja oko cene memorije i nejakih platformi više odavno nema

Uprkos svemu „Microsoft" je ipak otišao veoma daleko sa razvojem programa *Windows* rukovodeći se pre svega logikom da to treba da bude program koji će u kasnijim fazama razvoja raditi na svemu što se zove računar. Dakle, mi kao korisnici ćemo uvek i svuda bez obzira na mašinsku podlogu videti iste stvari i jednako lako se snalaziti na bilo kojoj mašini. Ideja jeste dobra, nimalo laka za realizaciju, ali toliko obećava da je teško ostaviti je. Uzgred takav stav objašnjava onoliko diskete kod je lako napisan da traži mnogo prostora, a to je posledica univerzalnosti pristupa

FORSIRANJE OS/2

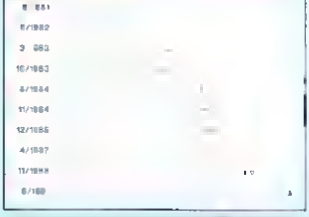
Ono što „Microsoft" iz sve snage razvija *Windows*, onda nije ima snage da se bog zna kako posveti razvoju OS/2 niti ima interesa da ga razvija jer bi ispalo da razvija sopstvenu konkurenciju. Velike firme su u principu strašno alave – one neće samo deo kolača nego celu tortu. Ovo se posebno odnosi na „Microsoft" koji se tokom prekle dekade navikao da ima celu tortu nazvanu DOS

Zbog ovakvog stava „Microsoft" i IBM su se rastali početkom ove godine – kako se pričа, ne baš mnogo prijateljski. IBM je najavio dalji ubrzani razvoj OS/2 a „Microsoft" razvoj sistema poznatog pod skraćenicom NT ili „Nove tehnologije" (*New Technology*). Taj novi sistem bi trebao da obezbedi univerzalnost po platformama i linearni pristup svim resursima mašine (to jest potpuno odbacivanje bilo kakvih barijera na 640 KB u Evropi verzija sve do 16 MB)

Sa svoje strane, IBM je počeo da radi forsirano na razvoju OS/2. Ono što se malo shvata jeste značaj OS/2 za IBM značaj koji je blago rečeno kritičan. Razloga za tu tvrdnju ima na pretek

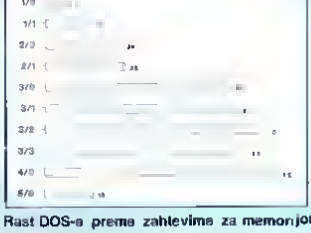
Prvo. Iržišno učešće IBM u segmentu ličnih računara je u stalnom padu, mahom iz dva razloga. Prvi je da se Iržište ličnih računara na Zapadu trenutno suzava usled recesije u SAD i veoma mutne situacije u Evropi odnosno ne sasvim jasnih postavki EEZ (ovo istina pogađa i sve druge a ne samo IBM). Drugi razlog je sve jača konkurencija, pre svega po pitanju cena i standarda. IBM je po definiciji skuplji od mnogih drugih kudikamo jeftinijih klonera pa tako zadržava segment Iržišta velikih firmi pre svega onih koje već imaju IBM-ove velike mašine, sve više gubi segment malih firmi koje u masi čine pozamašan deo tržista

Slika 1



Rast DOS-a preme novim verzijama

Slika 2



Rast DOS-a preme zahtevima za memorijom



ADUTI U RUKAVU?

Drugo velika uzdanica IBM-a mikrokanal nije postigao ni izbliza onaj uspeh koji sa od njega očekivao. mnogi veruju da je razlog tome IBM-ova prilično žestoka politika licenciranja koja je zahtevala i otplatu starih investicija u arhitekturu PC-a. Iako nema spora oko toga da je ova sabirnica tehnički daleko ispred današnje ISA sabirnice, postalo je jasno da za korisnika pojedinca nema bitnih razlika između ISA, EISA i mikrokanala – protok („throughput") je veoma sličan.

Treće, iako se najavljivao rat cena, pokazalo se da IBM ipak nije spreman (ili sposoban) za jedan tako drastičan korak. Jesu spustili cene, ali ne toliko da bi ugrozili čajvanski industriju. Na kraju krajeva i ovi ostali su snizili cene – morali su, pred najezdom sve jeftinijih klonova sa Istoka, koji su za malo postajali sve bolji i bolji, sustižući tako velika tehnološka imena.

Četvrto, IBM već poodavno nije na čelu kolone tehničkog napretka – pre bi se reklo da su na začelju. bar u slučaju ličnih računara. mada su imali značajnih uspeha (recimo, plazma prikaz na njihovim računarima se priznaje kao najbolji u svojoj klasi). Sem novog grafičkog standarda XGA, IBM nema nekih novotarija koje može da polegne kao adut iz rukava, ili se bar za njih ne zna.

Sve u svemu, jedino što IBM potencijalno ima jeste OS/2. Sada kada je njegov razvoj uzeo u svoja ruke (to jest, kada su se rastali sa „Microsoftom") razvoj OS/2 je naglo ubrzan. Što se vidi po sve češćim beta verzijama. Naravno, misli se na OS/2 verziju 2, koja bi trebala da bude nalik *Windows*-u, ali sa nizom prednosti.

RADIKALNE NOVOSTI

Srećnici koji su probali beta verzije kažu da se radi o zaista radikalno novom upravljačkom sistemu, sa veoma opipljivim prednostima, od kojih se navode sledeće:

- probijanje barijere od 640 KB i njeno prinudno presaljenje na 16 MB (zbog procesora 286) u verziji 2, odnosno 64 ili 96 MB u verziji 3, namenjenoj isključivo 32-bitnim mašinama
- grafički prislup okolini sa prozorima, za koji se kaže da je vidno bolji od onoga što nudi *Windows*, kao i potpuna kompatibilnost sa *Windows*-om, bar za sada (treba očekivati da će „Microsoft" veoma brzo otkloniti mogućnost kompatibilnosti, jednostavno zato što mu to ne ide u račun)
- podrška za više paralelnih operacija („multitasking") i više korisnika („multiuser"). Navodno i ovo veoma dobro radi.
- potpuno nov, do sada nikada viđen sistem rada sa datotekama koji, opet kažu, radi fantastično.

Što je posebno važno, svi izveštaji tvrde da OS/2 bez greške radi sa programima pisanim za DOS u jednom ili više prozora. Ovo je ujedno i *sine qua non*.

Problem razvoja oba sistema obuhvata pre svega pitanje kompatibilnosti sa starim DOS-om. Ovo je verovatno najznačajnije pitanje uopšte, jer niko sebi ne može da dozvoli luksuz odbacivanja svega što je bilo dotle. Uzmila samo da sa vrednosi programske baze proce njuje na preko USD 1 000 milijardi (svi proda programi do sada) – ko sme sve to lek lako da odbaci? Ko će kupiti novi upravljački sistem koji poliše sve što čovek ima? Niko. Dakle, ako OS/2 zaista ispuni ovaj uslov, tržište je njego vo.

KORIST ZA KORISNIKE

Najzad, potoji tu i jedno principijelno pitanje. Ako se *Windows* razvija za niz platformi kao univerzalni program, on će hteo-ne hteo morati da sadrzi i niz kompromisnih rešenja namatnutih različitim arhitekturama na kojima će raditi OS/2, sa svoje strane, treba da radi na „Intel" platformama, čime mu se sužava polje konsnosti se jedne strane, ali sa druge strana ovo mu omogućava visok stepen optimizacije koji *Windows* teško da moze dostići a kamoli prestici.

Iz svega ovoga postaja jasno da su oba sistema od životne važnosti za njihove autore. „Microsoft" bi mogao opet da postane najveće softverska kuća na svetu (sada je drugi iza „Computer Associates"), bez većih granica rasta. IBM bi mogao opet da postane voda, posle više godina kaskanja za drugima. Obe firme imaju velika resurse, mada naravno IBM ima veće – najveće na svetu.

Iako je tačno da obe firme mogu dobro da žive i bez svega ovoga, radi se o strateškim proizvodima, koji maltene garantuju tehničku prednost nad svima drugima. No, razlika ipak postoji – „Microsoft" ima nekoliko veoma uspešnih proizvoda, a IBM nema ni jedan izrazili proizvod koji vodi na tržištu.

Nezahvalno je prognozirati, ali mi ipak tipujamo na IBM. Njima je OS/2 zaista veoma potreban kao strateški proizvod, a sem toga, kao stvar principa – vreme je da neko malo zauzda pnlično haotični „Microsoft", koji pokušava da sedi na nekoliko stolica odjednom. Ako iko to može, IBM je taj.

U krajnjoj liniji, ma ko dobio bitku, mi kao konsnici u svakom slučaju dobijamo jedan bolji i daleko savremeniji upravljački sistem nego što je već potpuno otinjali DOS.

Softverske alatke

386 ALI DO MAKSIMUMA

"Qualitas Inc." firma-autor poznatog programa za upravljanje memorijom pod imenom *386-to-the-Max* inače glavnog konkurenta programu QEMM, počela je sa prodajom najnovije verzije, se oznakom 6, ovog programa. Prodaja je počela u dobrom znaku – program je na poslednjoj priredbi Comdex dobio jednu od dva nagrada za najbolji uslužni program uglednog časopisa *Byte*.

Nova varzija u odnosu na staru nudi pobolj šana performanse, ali i nekoliko potprograma koji su potpuno novi.

ROMsearch je novina koja pored celokup nih blokova traži i omogućava upotrebu i delo va naiskorišćenih segmenata ROM-a, čime se dobija veća količina memorije koja se moza korisno upotrebiti.

VGAswap praktično realocira ROM EGA i VGA video karti, čime se dobija veća količina gornje (640-1 024 KB) DOS memorije, u koju se potom mogu preseliti razni uslužni i/ili kori snički programi.

SmartMemory oslobađa korisnika razmišljanja kakva vrsta (produžena ili proširena) memorije mu je potrebna za ovu ili onu aplikaciju. Potprogram sam odreduja optimalne od nose i rasporede korišćenja, a rezultati su lak ša instalacija, racionalnija upotreba i nešto po vecana brzina rada.

Maximized Reordering je proces upotrebe različitih oblika memorije i rasporeda veznih i pobudnih programa, kako bi se, s jedne strana oslobodilo što više memorija, a sa druge strane dobio željani raspored prebacivanja iz DOS u druga vrsta memorija. Programi se sređuju tako da se prilikom podizanja sistema na optimalan način prebace u potrebnu vrstu memori ja.

BIOS Compression je metod kojim se na PS/2 sistemima dobija i do 250% više slobodne gornje DOS memorije, što se postiže uklanjanjem naredbi iz PS/2 ROM-a koja su nepotrebne u radu pod DOS-om. Ovim se oslobada dopunskih 80-64 KB gornje DOS memorije.

Verzija 6 u celosti podržava DPMI (DOS Protected Mode Interface – rad sa zaštićenom memorijom pod DOS-om), pristup koji prihvata sve veći broj autora programa (recimo *Lotus 1.2.3 3.1, Borland C++, Microsoft C 6.0* i na ravo *Windows*). Ovaj režim je namenjen 32-bitnim procesorima – ako na želite da cekate DOS 6.0, evo prilike da ga malo osetila u prak si.

Tu je i potpuno nova verzija (2.0) programa *Asq*, koji prikazuje stanje slepen i malode is korisconosti memorije u računaru. Moguce je napraviti i sliku (*snapshot*) raspoređenosti memorije, koja se neposredno iz ovog programa može poslati na štampač, kako bisle imali tak uvid u stanje stvari kada god to poželite.

Najzad, *386 to-the Max* od ove verzije može da radi sa celom porodicom procesora firme Intel, od 8086/8088 do 80486, uz uslov da u hardveru podržavaju duplikalnu memoriju (Shadow RAM) i/ili LIM 4.0, ili da koriste Chips & Technologias NEAT (286/386) odnosno LEAT pakete čipova.

Cena nadogradnje na verziju 5.2 je USD 29.95 + USD 5 za poštarinu. *(D.V.)*

Konsrta adresa Qualitas Suite 1386 7101 Wisconsin Avenue Bethseda MD 20814 S.A.D. Tal 991-800-676 6386 Fax 991 301-718-6061

PROZORI ZA VISOKU KLASU

"Microsoftove" strstegije u obiasti operetivnih sistema može sa opiseti samo jednom rečju: Prozori! Uspeh bez presadana *Windows-a 3.0* (u prvoj godini isporučeno je četiri miiiona kopija, a u ovoj se očekuje prodaja dodatnih osam miiiona) još viša će učvrstiti ovu strategiju u budućnosti. Krunu te strategije treba de predatavlja *Windows NT*. On objedinjuje nekoliko operetivnih sistama i može se izvršavati na raznorodnim herdverskim platformama.

Kade je *Windows* izbačen na softverako tržište velike većina korisnika redila je na stonim PC računerima U meduvremenu, herdverska osnova sa proširila Današnji korisnici mogu de rade na bilo kojoj kombinaciji portebl i kućnog računara dok izvesten manji broj korisnika i RISC radne stanice i servera Cilj „Microsofte" je da investicije u nove svetske softverske standarde, kao i u DOS i *Windows* održi ali i de ih poveća ne sprečavajući korsnike da istovremeno prate i napredak hardvera Drugim rečima Microsoft" želi de slavi na raspolagenje različite implementacije jednog jedinog. fleksibilnog operativnog sistema – *Windows* – prilagodijivog raznim hardverskim platformema

DVE ŠIROKE KLASE

Trenutne verzije *Winodws-a* kao i one koje „Microsoft" tek planira da stvori, spadaju u dve široke klase

- U prvoj su proizvodi koji sa baziraju na sadešnjem i budućim verzijama DOS a i *Windows-a* Ove *Windows* implementacije biće proširene na način da mogu u potpunosti da koriste pogodnosti procesora „Intel" 386SX/386/486 i nudiće funkcije koje će zadovoljiti najveći broj „desktop" korsnika One će biti tako oblikovane da budu direktna dopuna DOS-u Ove strategije je već potvrđena u praksi
- Druga klasa će obuhvatati fleksibilne „high-end" implementacije *Windows-a* sa istim korisničkim interfejsom i istim modelom programiranje, i sa mogućnošću izvršavanja svih aplikacija napisanih za DOS verzije Ove verzija nazvana *Windows NT (New Technology)* ponudiće poboljšane funkcije operativnog sistema, koje će ispunjavali visoke zahteve u sigumosti podataka, ze servere visokih performansi enažne radne stanice i klijent/server sisteme

Obe verzije operetivnog sistama *Windows* će se isto vladeti prema korisniku – imaće isti model programirenja i bazirače se na fleksibilnoj arhitekturi, koja će kako korisnicima lako i proizvođačima omogućili da svoje postojeće *Windows* investicije i specifične znanje upotrebljavaju i dalje keda predu na novu verziju operativnog sisiema Korisnicima DOS-verzije *Windows-a* biće odslupan i korisnički interfejs high-end verzije Tako će proizvođači softvere ponudili jedan jedini programski interfejs (API) koji će biti ugrađen u sve *Winodws* implementacije i koji će dozvoljavati korišćenje širokog spektra „Intel" i RISC platformi *Windows-32 Bit API* sadrži u sebi i trenutne DOS- i *Windows 3.0/3.1* API-je i ponaša se naravno kao puni 32-bitni API koji se bazira na sedašnjem API ju *Windows-a*

Nezavisni isporučioci softvera veoma mnogo investiraju u aplikacije namenjene „Microsoftovom" *Windows* u U velikom je interesu i njihovom i njihovih kupece da se te investicije što je moguće brže isplate kao i da se obuhvate nove hardvereke platforme *Winodws 3.0* je trenutno na raspolegenju za dve platforme (za koje se isporučuje jedinstveno) *Windows* u standardnom režimu za 286-ice sa 1 megabajtom memorije i *Windows* u proširenom režimu za 386 ice sa najmanje 2 megabajta memorije Obe verzije za obe platforme su direktne proširenje DOS a i nude multitasking integraciju posiova i grafički korisnički interfejs (GUI) Prošireni režim podržava i straničenje memorije (paging) i višestruke DOS prozore

NAJNOVIJI MODEL: *WINDOWS NT*

"Microsoft" je, u meduvremenu, pripremio novu high-end verziju *Windows-a* koju će isporučili korisnicima već ove godine – *Windows NT* To je odvojena implementacija predviđena za mašine visoke klase počev od 386 486 i boljih „intelovih" procesora sa 8 megabajta memorije kao i za RISC računere i servere *Windows NT* je zamišljen kao dopuna a ne zamena za DOS-*Windows* On je kamen temeljac „Microsoftove" „New Technology" filozofije nova osnove operativnih sistema sledeće dekade

Windows NT je samosalnu operativni sistem se potpunom emulacijom DOS a i svim funkcijama trenutnih implementacija *Winodws-a* On sadrži i napredne tahnike kao što su podrška simetričnom multiprocesiranju podrška 386/486 platformi i MIPS RISC-a multitasking u raspodeljenom vremenu garantovane sigumost sistema veome moćan U/I (ulaz/izlaz) visokih performansi integrisane mogućnosti rade u mreži puna podrška radu kako sa 16 tako i sa 32 bita Sistem je predviđen kako za potrebe klijenta tako i za potrebe servera i radi zajedno sa „Microsoftovim" *LAN Manager*-om U svojoj prvoj verziji će *Winodws NT* pored DOS i *Windows* aplikacija podržavati takođe i POSIX i važne OS/2-Server-aplikacije

Windows NT igra ključnu ulogu u „inicijativi za napredno računarsko okruženje" (Advenced Compuler Environmenl Initiative – ACE) koju su u aprilu 1991 zajedno obnarodovali „Microsoft" „Compaq Computer" „Digital Equipment" „MIPS Computer Systems" „The Santa Cruz Operation" i drugi Danas ove inicijative okupija više od 200 članova.

Cilj ACE-inicijative je da jedno otvoreno napredno okruženje bazirano na standardima slave na raspolaganje za sisteme ne baz raznih mikroprocesora ACE inicijativa nudi punu podršku dvema platformama PC-sistemima na bazi „Intel" 386/486 procesora i sistemima ne bazi MIPS RISC a Kao prenosivi (portabilni) operativni sistem *Windows NT* je presuden elemenl ACE standarda

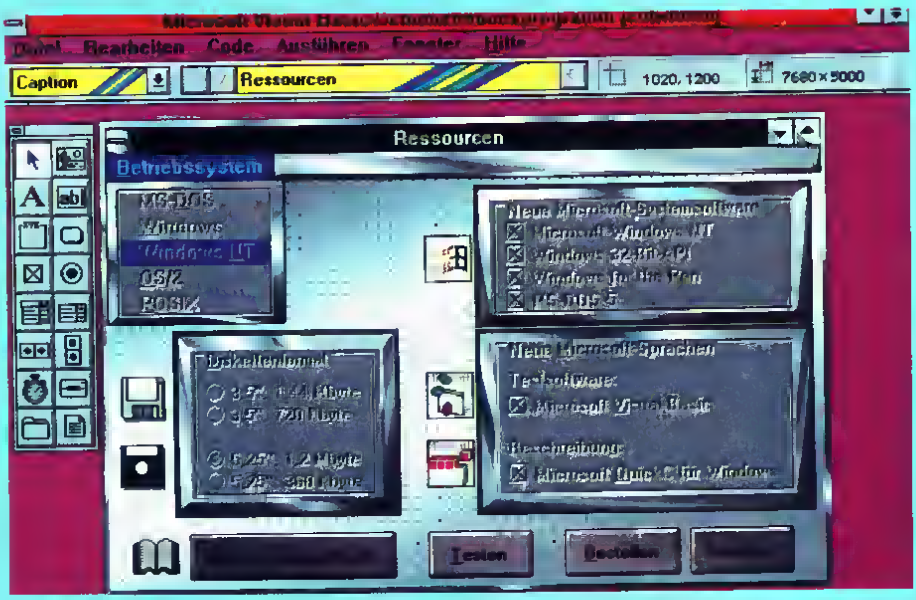
OSNOVNI PRINCIP I KOKENI

"Microsoft" je svoju nemeru da proizvede prenosivu verziju svoje familije high end operativnih sistema obznanio krajem 1989 dok je u januaru 1991 otkrio nove pojedinosti štampi i analitičarima Ali projektovanje i stvarni rad zapravo su počeli već 1988 U tom periodu je tim programera sa iskustvom u razvoju operativnih sistema pošlavio temelje sistema koji je „Microsoft" uzeo kao bazu za devedesete Tim je predvodio Dejvid Kaller (David Cutler) jedan od očeva široko rasprostrenjenih operativnih sistema RSX i VMS Osiali članovi tima imali su veliko iskustvo u razvoju MS-DOS *Windows* VMS UNIX i OS/2 operativnih sistema

"Microsoft" je došao do zeključka de high-end operativni sistem devedesetih mora ispunjavati opšte zehteve u funkcionalnosti kao što je ne primer fleksibilna podrške raznorodnog hardvera visokih performansi Ali on u principu mora biti i proširljiv Pod ovakvim operativnim sistemom ne treba de se izvršavaju samo aplikacije koje sa prodaju u velikom broju primeraka (koje će uvek u obilju biti razvijene za *Windows*) već i aplikacije sa melim brojem prodatih kopija pisane za POSIX i OS/2 Dalje sistem mora dozvolili buduće izmene na primer proširenje svoje funkcionalnosti u oblasti podataka i integracije poslova ne bazi objektnih tehnika (od kojih se očekuje da će u devedesetim igrali značajnu ulogu u računarskom svetu)

Windows NT ime u neku ruku slojevitu strukluru Osnovne funkcije operativnog sistema su na raspolaganju u osnovnom nivou (koji je kod *Windows-a NT* nazvan „Executive") i one podržavaju aktivnosti zaštićenih podsistema (nazvanih isto serverima) Zaštićeni podsistemi koji se izvršavaju kao standardne aplikacije u korisničkom režimu aktiviraju pomoćne programe (utilities) koji su karakteristični za svaki operativni sistem ponaosob implementiraju se one funkcije operativnog sistema (okruženja i API-ji), koje korisnički programr očekuju Budući da istovremeno može biti aktivno više ovih zaštićenih podsisteme može se izvršavali više različitih tipova korisničkih programe istovremeno U slučaju operativnog sisteme *Windows NT*, „Microsoft" će stavili na raspolaganje zaštićene podsisteme za efikasnu podršku *Windows* DOS POSIX i OS/2 programe Bilno je primetili de *Windows NT* očuveva svoju fleksibilnosl bez kompromisa u performansama

Pri projektovanju *Windows-a NT* uzela su u obzir i iskustva „Microsoftovih" programera koji su ne postojećim sistemima implementirali *Microsoft LAN Manager* nastojeći da mu performanse budu jednake ili bolje od namenskih file server sistema Normalno de je jednom operativnom sistemu opšte namene vrlo teško da ponudi isle performanse kao neki specijalizovani sistem Ova iskustva preuzele su pri projektovanju „kostura" *Windows NT*-a Ovim neće samo profitirati „Microsoft" *LAN Manager* već i drugi mrežni sistemi a easvim uopštano i U/I operacije



KOMPONENTE SISTEMA

Operativni sistem *Windows NT* sadrži četiri klase komponenata. U privilegovanom sistemskom režimu (sa direktnim pristupom hardverskim resursima i zaštićenim funkcijama operativnog sistema) izvršavaju se sledeće komponente

- **Executive** stavlja na raspolaganje osnovne funkcije operativnog sistema koje potrebne drugim komponentama
- **Proširenja privilegovanog režima** u koja spadaju drajveri uređaja (device drivers) sistemi datoteka (dalje ćemo ih zvati file-sistemi) i procesi (na primer *LAN-File Server*)

U neprivilegovanom (ili korisničkom) sistemskom režimu izvršavaju se kao posebni procesi (i konsole usluge privilegovanih komponenti sistema) sledeće komponente

- **zaštićeni podsistemi** koji stavljaju na raspolaganje rutine (*utilities*) za razne korisničke programe
- **korisnički programi**

Ove komponente ćemo sada detaljnije opisati

FUNKCIJE EXECUTIVE-a

Executive je arca ovog sistema. Zahvaljujući njemu *Windows NT* može u isto vreme da podržava neki *Windows* podsistem koji ima neku svoju strukturu procesa i neki POSIX podsistem koji ima neku sasvim drugačiju strukturu procesa. Kod većine ne-UNIX operativnih sistema je teško ako ne i nemoguće efikasno podržavati POSIX programe.

Najvažnije osobine *Executive*-a su

Microkernel kompaktni *microkernel* (oko 50 K kompajliranog koda) koji je ugrađen u *Executive* implementira mašinski zavisne funkcije najnižeg nivoa sinhronizaciju multiprocesinga obrednu niti (threads) i kernel-objekte. Ovde se nalazi kod koji se mora izmeniti ako se sistem želi prebaciti sa jedne mašinske arhitekture na drugu. Shodno tome, rad potreban da se *Windows NT* prebaci na novu hardversku arhitekturu je mali.

Usluge (utilities) na višem nivou Ostalek *Executive*-a koristi kernel funkcije i stavlja preostale opcije sistema na raspolaganje na višem nivou. Ovaj deo koda (koji je glavni deo *Executive*-a) se u potpunosti koristi u respodeljenom vremenu i ima više niti. Ovime ne protiči samo aplikacije na sistemu se više procesora, već i sam operativni sistem. Operativni sistem se može izvršavati na jednom ili više procesora odjednom.

32-bitne zaštićene etrenične memorije (32-Bit-Protected Peged Memory) Sistem koristi 32-bitni memorijski model i sadrži potpunu zaštitu memorije za svaki proces. Inače bi neke aplikacije mogle srušiti ne samo neku drugu aplikaciju već i sem sistem. Sistem podržava mapiranje datoteke u virtuelnoj memoriji i jedinstveno straničenje ovih datoteka sa U/I sistemskim keširanjem (I/O-System Caching). Nisu potrebne nikakvi suvišni baferi da bi se podaci iz adresnog prostora neke aplikacije prebacili u datoteku na disku ili na neki drugi periferijski uređaj.

Prenosivost *Windows NT* je prvobitno razvijen za Intel 386/486 familiju a zatim je prenet na MIPS RISC-arhitekturu. Uopšteno gledajući sistem se može brzo i efikasno preneti i na druge arhitekture.

Simetrija Kao što je već ranije spomenuto *Windows NT* je od svog osnova pa naviše tako segradien da podržava simetričan multiprocesorski hardver i tako omogućava veliku fleksibilnost sistemskih performansi. Pomoću simetričnog dizajna izvršavanje niti (threads) biće raspodeljeno postojećim procesorima u hardveru. Posledica ovoga je da je i operativni sistem brži a aplikacije se više niti biće brže ako se izvršavaju na multiprocesorskom hardveru.

Sigurnost *Windows NT* je siguren i zaštićen sistem. Podržava red sa više korisnika, što će vam biti potrebno ako ga koristite kao mrežni ili terminalski server. Prilog sigurnoj raspodeljenoj obradi u *Windows-u NT* je njegova mogućnost da dozvoli serveru da iz bezbednosnih razloge „personifikuje" svoje klijente. To rešava jeden od problema koji se javljaju kada želimo da od opšteg operativnog sistema napravimo dober operativni sistem na raspoređenoj mreži. Problem je u sledećem: da bi se obezbedila sigurnost na višem nivou neophodno je garantovati da se bezbednosne provere obavljaju na jednom i samo na jednom mestu – u operativnom sistemu. Iako da ne postoje nikakva druga „mala vrata" kroz koje bi se mogio provaliti u sistem. Ne žalosti serveri uglavnom rade tako da na zahtev udaljenog klijenta moraju njega servisirati i stoga su u normalnim prilkame tako strukturirani da je sam server nadležna i privilegovana jedinica koja sve može uraditi. To znači da server more sam odlučiti da li će određene neloge klijenle proslediti dalje. To automatski dovodi do toga da se odluke vezane za sigurnost ne donose na jednom, već na dva masta u sistemu – u serveru i u osnovnom operativnom sistemu. *Windows NT* to izbegava: pošto dopušla serveru da brzo na sebe preuzme sigurnosni protil svoga klijenta kada po njegovom nalogu preduzima akcije. Tako zahlevi služu praktično direktno od klijenta a provere se vrše samo jednom unutar operativnog sistema.

Jeene unutrešnje struktura Jedne od metoda kojom *Windows NT* postiže sigurnost u radu i ostvaruje mogućnosl proširenja sastoji se u upotrebi jedne standardne šeme u okviru koje se upravlja svim resursima sistema kao što su datoteke proceel semafori penjenje i

drugo. Ovakve objektna arhitekture stavlja nam na raspolaganje trajan, dosledan skup semantičkih previle i operacija za rukovanje svim objektima u sistemu.

Skupovi znakova *Windows NT* podržava 16-bitni Unicode skup alfanumeričkih znakova koji sadrži sve svetske azbuke i abecede. To znači da sistem *Windows NT* može bilo gde u svetu biti upotrebljavan na lokalnom jeziku (na primer pri imenovanju datoteka menja ild) *Windows NT* će omogućiti i da autori korisničkih programe lako svoje programe oblikuju tako da mogu da rade sa bilo kojim svetskim jezikom i da se iz bilo kojeg jezika mogu lako i brzo prevesti na bilo koji drugi svetski jezik. Ovo važi i za jezike sa potpuno različitim strukturama azbuka (abeceda) što je na primer slučaj sa evropskim i dalekoistočnim jezicima.

Važno je napomenuti da će tunkcije *Executive*-a koristiti i svim ostalim komponentama sistema. Tako se, na primer, mehanizmi za sigurnost i višeprocesorski rad koji su ugrađeni u *Executive* mogu prenositi i na sva proširenja i aplikacione programe.

PRIVILEGOVANA PROŠIRENJA

Proširenje sistema mogu se ostvariti preko programa za direktan pristup hardveru ili programa koji zbog boljih performansi moraju biti izvršavani u privilegovanom sistusu mašine. U privilegovana proširenja spadaju

Drajveri periferija (device drivers) To su biblioteke koda (code libranes) koje stavljaju na raspolaganje usluge periferija kao što su disk i mrezni U/I. Na primer, *Windows NT* će imati generički SCSI drajver koji će delovati preko manjih specifičnijih drajvara za različite SCSI adaptere. To znači da će pisanje jednog drajvera biti znatno pojednostavljeno. To će, na primer olaksati da se TCP/IP drajver sa UNIX-e prebaci na *Windows NT*. Drajveri periferija mogu dinamički biti učitavani u sistem, čime se na jednostavan način omogućave proširenje *Windows NT* sistema za vreme dok se on izvršava.

File sistemi Uopšteno gledajući posloji visok stepen podudarnosti između file sistema i drajvera periferija. File sistem je drajver periferija, all na jednom višem nivou. Kao i kod drajvera, i ovde veliki broj pripremljenih funkcija *Executive*-a olakšava podršku novih file sistema. „Microsoft" će obezbediti barem ove vrste file sistema za *Windows NT*: DOS-kompatibilni *File Allocation Table* sistem (FAT), OS/2-kompatibilni *High Performance File System* (HPFS) i neki file sistem za CD ROM.

Nove funkcije Objektno orijentisana struktura *Executive*-a omogućava da se nova funkcije direktno pridodaju *Executive*-u bez briga da sistem time može biti omeian. Očekuje se da će ovo u budućnosti biti od velikog značaja.

Privilegovani procesi Obično se želi da se velika i kompleksna proširenja sistema napišu kao regularni programi sa pristupom celokupnom programerskom okruženju. Tako ih je moguće lakše razvijati i debagovati. Zbog performansi, ove funkcije (programi) moraju da se izvršavaju šio je moguće više u privilegovanom sistemskom režimu. *Windows NT* to ima u vidu. Na ovakav način će, na primer, biti implementiran programski kod file servera *LAN Manager*-a. Njegov rad u privilegovanom režimu znači da će vremenski „skupa" promena statusa biti izbegnuta, što znači brži odgovor na zahteve koji dolaze iz mreže. Ovo je veoma opšta arhitekture i moći će da je koriste i proizvođači drugih mreža.

ZAŠTIĆENI PODSISTEMI

Najjednostavnija metoda da sa napišu velika i vazne proširenja sisteme je da se ona pišu kao regularni programi za koje imamo sve funkcije i alate za debagovanje. Pošto sistem ostaje i robusian i lako ispunjava zahteva sigumosti ova proširenja se mogu izvrsavati i kao regularni programi i u neprivilegovanom režimu. Ovakva neprivilegovana proširenje u *Windows NT* žargonu će se nazivati „zaštićeni podsistemi". Kada nekom konsničkom programu (klijent) koji je u toku zatraba mu neka sistemske funkcija on upućuje zahtev za tom sistemskom funkcijom, taj zahtev prihvata operativni sistem koji ge onda direkino prenosi „lokalnom operativnom sistemu", koji će biti nazvan „zaštićeni podsistem" (ili server). Ovaj mehanizam naziva se još i RPC (*Remote Procedure Call*) između kljenla i servera. Podsistem se izvršava u zasebnom adresnom prostoru, odvojen i potpuno zaštićen od klijenle.

Microsoft" namerava da razvije tn ovakva podsisteme.

Neka svojstva 32-bitnog Api-ja

Asinhroni unos

Za razliku od *Windows*-a 3.x i *OS/2 Presentation Manager*-a 32-bitni API radi multitasking u raspodeljenom vremenu. Is to tako. ako neki program očekuje podatak preko tastature ili miša. Io konsnika ne sprečave da radi sa ostalim programima.

API za konzole

Pomocno sredsivo za znakovno orijentisane programe koji se izvršeivaju u nekom prozoru.

Unikod

Unikod (Unicode) je višejezični skup znakova koji pojednostavljuje razvoj nacionalnih programa.

Semafori

U Windows-u NT postoje razliciti tipovi semalora.

Niti (Threads)

Unutar jednog procesa može biti aktivno više niti!

Virtuelna memorija

Program ima uticaje na dodelu virtuelnih stranica memorije, kao i na obezbeđivanje velikih povezanih virtuelnih adresnih prostora.

LAN

Ugrađene pomocne funkcije za računarske mreze na primer cevi (pipes) sa imenima, pozivi funkcija preko mreze (RPC) raspodela poruka (Mail Slots).

RAM datoteke

Program moze datotekama ne hard-disku de pristupa kao da se nelaze u memoriji. Sistem brine o pranosu podataka izmedu datoteka i memorije.

Bezierove krive

Grafički interfejs (GDI) je proširen Bezierovim krivema.

Bitmap funkcije

Nove funkcije za rasterske slike omogucavaju rotaciju slika i mogu izvršavati kompiikovane operacije sa maskama.

Promena koordinata

16-bitni API kod *Windows*-a može transliran koordinata a može ih i skalirati (ZOOM). 32-bitni GDI moze translirati i objekte.

Staze

Staze nude pomoć pri razvoju kompleksnih jednodimenz onih (linije) i dvodimenzionih (ispunjenih) geometrijskih ligura. Moguće je i opsecanje tigure (*clipping*) po željenoj stazi.

Windows podsistem

Windows podsistem će obezbediti sistemske funkcije (uključujući i manipulaciju prozorima i grafiku) koje će biti potrebne 16 bitnim i 32-bitnim *Windows* programima i DOS programima. 32-bitni *Windows* programi bice programi koji vertovani tako da podrzavaju prošireni 32-bitni API. „Microsoft" je sve potrebne specifikacije uslupio vodecim nezavisnim softverskim kućama o namerava i da ih objavi. Ovaj podsistem moći će da emulira x86-CPU tako da će se 16-bitna DOS i *Windows* aplikacija moći da se izvršavaju i na MIPS CPU. „Microsoft" je u saradnji sa vodećim svetskim *Windows* softverskim kućama definisao novi 32-bitni API.

Signal za 32-bitnim API-jem došla je iz softverskih kuća koja su želele da zadrže dobro poznat *Windows* model programiranja i *Windows* izgled ali i da konsle prednosti pravog 32-bitnog memorijskog modela. Zato je i prvi cilj pnilikom definisanja *Windows* 32-bitnog API-ja bio da se proces konverzije 16-bitnih *Windows* programa na 32 bita učini što je moguće lakšim. Osim toga, softverska kuće su nabrojale i druge funkcije i usluzne programa koja su želele da vide i u 32-bitnom API-u (na primer Bezierove krive, komunikacija između procesa imenovane cevi (pipes) semafori i drugo). API je proširen dodatnim novim funkcijama za obradu podataka za semalore ze visestruke niti a značajno mu je proširen i grafički interfejs (vidi tabelu).

Windows se još uvek oslanja na DOS pri otmrenju otvaranju i zatvaranju datoteke i njihovom čitanju i pisanju. Iako on ima par 16-bitnih funkcija koje ovaj posao, po svoj prilici mogu obaviti (na primer **OpenFile**, **_lread** i **_lwrite**) ova funkcije zapravo nisu ništa više od obloge za DOS funkciju **Int21h** koja zapravo obavlja posao. U 32-bitnom API-ju imamo jedan potpun skup funkcija za rad sa datotekama. Otvaramo datoteku sa **OpenFile**, kreiramo datoteku sa **CrsateFile**, čitamo iz datoteke sa **ReadFile** i u nju pišemo sa **WriteFile**. Druge funkcije nam omogucavaju pristup atnbulima datoteka, na primer **GetFile** nam daje podatak o veličini datoteke a **SetFileTima** menja podatak o lome kada je datoteka zadnji put obrađivana. Nove funkcije pretražuju direktorijum (**FindFirstFile** i **FindNextFile**), kopiraju datoteke (**CopyFita**) ili ih brišu (**DeleteFile**). Datoteku možete premestiti (**MoveFile**), ili zaključati neke njene delove (**LockFile** i **UnlockFile**). 32-bitni API može sleviše preslikavati datoteke u memoriju. Na ovaj način može neka datoteka koja se nalazi na hard disku biti dodeljena domenu virtuelne memonja. Sadržaj datoteke se može jednostavno pročitati ili upisati, budući da program pnstupe samo segmentu memorija. Datoteke ostaje na disku sve dotle dok joj se stvarno ne pristupi. Ako dođe do pristupa a traženi dao datoteka nije smešten u memoriji on će biti (što sam program ne vidi) učitan u memoriju.

Ovekve RAM datoteke (*memory mapped files*) veoma pojednostavljuju rad sa datotekama, jer kroz njih memonjski prostor na disku postaje proslo proširenje RAM memonje. Budući da sistem učitave samo traženi segment RAM datoteke, ona je idealna za rad sa ogromnim datotekama, na primer sa velikim bazama podataka.

Toliko o ovom 32-bitnom API-ju. Sve u svemu, *Windows* 32-bitni API je potpuni, robustni i visokorazvijani API pogodan za poslove sa visokim sigurnosnim zahtevima i za aplikacije na radnim stanicama, dok istovremeno zadržava fundamentalnu kompatibilnost sa postojećim *Windows* modelom programiranja. 32-bitni API najpre će biti ugrađen u *Windows NT*, a kasnije i u neku buduću verziju *Windows*-a baziranog na DOS-u.

POSIX podsistem

POSIX podsistem će izvršavati programe koji budu napisani za POSIX interfejs (IEEE 1003.1). POSIX je specifikacija sistemskih funkcija koja u potpunosti sledi UNIX-model.

OS/2 podsistem

Ovaj podsistem podržavaće programe koji budu napisani po meri OS/2 verzija 1.3 i 2.0. Prva verzija *Windows NT*-a podržavaće nepromenjene 16-bitne OS/2 serverske aplikacije, kao što je na primer SQL server. Kasnije verzije ponudiće punu podršku OS/2 aplikacija.

CILJ: KOMPATIBILNOST

Fleksibilnost i proširljivost nisu dovoljne osobine jednog operativnog sistema, on treba i da radi veoma efikasno. *Windows NT* nudi efikasne metode za prenos informacija između korisničkih programa koji se izvršavaju i zaštićenih podsistema koji svoje usluge stavljaju na raspolaganje aplikacijama. (Na primer, ima veoma efikasan RPC mehanizam za međusobne pozive klijenata i servera.)

Da bi se postigla još veća efikasnost, podsistemi će biti pozivani na selektivnoj osnovi. U slučaju jednostavnih čitanja i pisanja pri radu sa datotekama, mogu se na primer funkcije iz *Executive*-a pozvati i direktno, bez poziva preko zaštićenog podsistema. Stoga će sistemi bazirani na *Windows NT*-u nuditi kako visoke performanse, tako i fleksibilnost i mogućnost proširenja.

"Microsoftov" cilj je da korisnicima omogući da svoje trenutne investicije u *Windows* aplikacije sačuvaju i kasnije kada eventualno pređu na nove hardverske platforme viših performansi. Zato je ova firma razvila jedan jedini prilagodljiv operativni sistem – *Windows* – sa kompatibilnim implementacijama za niz mikroprocesora. Trenutno ove implementacije obuhvataju verziju *Windows*-a za PC je bazirane na x86 procesorima i *Windows NT* za napredne „Intel" mikroprocesore i MIPS RISC mikroprocesor.

Windows NT je potpuno moderan i doleran sistem, na kome će se izvršavati najveći deo komercijalno raspoloživog korisničkog softvera i koji će očuvati investicije softverskih kuća i krajnjih korisnika. Obećana kompatibilnost trebalo bi da dovede do još većeg investiranja. Softverska kuće mogu i dalje da pišu *Windows* programe jer znaju da sada mogu ciljati na čitav spektar računara – od 386SX notebook računara pa do 486 i MIPS multiprocesorskih servera.

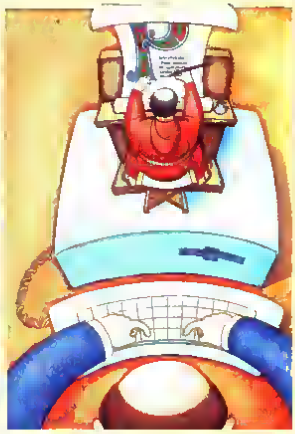
Za krajnjeg korisnika je značajno što će biti zadržan isti korisnički interfejs kod svih nadalje proširivenih DOS i *Windows* baziranih sistema i high-end sistema na bazi *Windows NT*-a tako da će moći da se koriste programi i tehnike na različitim platformama bez ponovnog „ugiravanja".

Prema „Microsoft System Journal"
Pripremili Kragi Aleksandar i Vuk Ivanović

SEDAM SMRTNIH GREHOVA

Ovde ćemo vam sa našag čednog i bezgrešnog stanovišta razotkriti moderna zla koja opsedaju današnje korisnike ličnih računara...

Čovečanstvo je preplavljeno grehom. Ako ne verujete, pitajte bilo kojeg propovednika. Pitajte bilo kojeg grešnog propovednika. Dok je Desel Zapovesti skup pravila pomoću kojih se odlučuje o vašoj sudbini na „onom svetu", dotle Sedam Smrtnih Grehova određuje vašu sudbu dok ste živi. Ovi uzasni svakodnevni gresi nisu iščezli, i progone nas do dana današnjeg. Zapravo, moderni korisnici računara nesvesni njihovog uticaja, možda na uvidaju dominaciju ovih graha u većini aspekata računarstva i ne uočavaju njihov uticaj na baksuznog (i naivnog) korisnika računara. Upoznajmo Davola da bismo ga više voleli.



Greh Broj Jedan: Lenjost

Malobrojni su računarski korisnici koji mogu izbeći ovom najuzasnijem od grehova. Kada ste zadnji put naterali sebe da procitate pratecu dokumentaciju uz softver? (Da ne spominjemo greh što dosta često vas softver i nema dokumentaciju.) Suviše ste lenji? Niste pomislili da bi to moglo biti važno? A šta bi sa čišćenjem onog musavog ekrana vašeg monitora? Kada ste zadnji put nagovorili sami sebe da uradite je dosadne poslove? Neki korisnici nisu bili voljni čak ni da utroše 15 minuta potrebnih za nadogradnju do poslednje verzije DOS-a pa i dalje koriste DOS 2.0 (Opravdanje „Zašto da se gnjavim sacekacu DOS 6.0!")

Greh Broj Dva: Škrtost, Pohlepa

Jeste li ikada išli kod prijatelja i tražili od njega piratske kopije programa za koje ste znali da ih nikada neće upotrebljavati? Želite kopije samo zbog toga što „morate imati jedan primerak!" Neka izgubljene duše vođene pohlepom provode sve svoje vreme vršeći download programa sa Sezam-a (a i drugih, manje popularnih mreža i BBS ova) i pored toga što ih nikada neće pregledati. „Još još još!" Pravi rezultat ove pohlepe je više novca potrošeno na telefonske račune.

Greh Broj Tri: Ljubomora, Zavist

Nervira vas prijatelj koji ima najnoviju 486 mašinu. Vi jakođe morate imati jednu takvu. Trebalo vam ili ne. Gubite miren san i novac.

Greh Broj Četiri: Nezasitost

Imate hard disk od 200 megabajta i uspeli ste da ga nakrcate sa 198 megabajta programa i podataka – uglavnom bezveznih. Napo znate datoteke otkačeni pomoćni programi, programi zanemarljive vrednosti. Ali možete li se odlučiti da izbrišete sve te stvari? „Možda ce biti važne jednog dana." Zbog toga kupujete još jedan disk od 200 megabajta.

Greh Broj Pet: Bes

Do ovoga dolazi usled aspekta gubljenja vremena u računarstvu. Utrošicete celo poslepodne ponovo štampajući jedan te isti dokument, pokušavajući da ispadne „baš kako želite" ili ćete provesti ceo dan na posiu hipnotisani Tetrisom na mogavši da prekinete. Kad stignete kući, šutnućete svog psa.

Greh Broj Šest: Požuda

Provodite dosta vremena okapajući nad računarskim časopisima, gledajući sve te poslednja i najnovije proizvode. Sanjate o njima. Pokušavate da ubedite svoje roditelje (bračnog druga? decu?!?) da vam dozvole da kupite nešto od toga. U stvari, sline vam cure nad oglasima u „Računarima".

Greh Broj Sedam: Ponos

Uživate gledajući opremu koju posedujete. Hvališete se svojim prijateljima. Objašnjavate im kako je vasa masina zaista brza u poređenju sa njihovom. Pokazujete im gomile računarskih časopisa i knjiga o DOS-u i pevate ode o tome kako računar poznajete „u dušu". Gievni ste u kancelariji. Kada neko nešto želi da sazna, vi mu kazete: „Šta o tome mislite bez obzira da li nešto znate o tome."

* * *

Na kraju, ipak reč-dve. Mali broj računarskih grešnika pati od svih navedenih grehova. To je skoro namerno. Ali vecinu nas pogađa barem jedan od njih. Zapravo, bolje je reći vecinu vas, jer ja nemam tih problema. Završiću sada ovo razmatranje jer želim da pokažem prijatelju gomilu raznih stranih računarskih časopisa iz kojih prevodim lekcije koje kucam na moćnoj najnovijoj i najboljoj 486 mašini. (Uzgrad sjajno i programiram – najodlji sam!) Uobraženost? Ala-vost? Nezasitost? Nema to ga kod mene, hvale lepo.

Prema PC Magazinu
Pnredio Aleksandar Kragi

SVETLOSTI I SENKE

U praistorijaka vramana, kada ja PC CAD bio ograničan na dvodimenzionalno iscrtavanje tehničkih crteža, nije bilo potrebe za vizualizacijom objekata. Prostorni modeli objekta, sa manje-više realističnom vizualizacijom, predstavljaju jednu od najvažnijih osobina 3D paketa. Kada je i *AutoCAD*, u varziji 2.6, dobio solidna 3-D mogućnoati, odmah sa pojavio i odgovarajući dodatni pakat za aančenja *AutoCAD* modela – *AutoShade*.

Nebojša Novaković

Mogućnosti prve verzija *AutoShade* izgledaju smešna u poređenju sa današnjim paketima za senčenje i vizualizaciju trodimenzionalnih modela. Prosto predstavljanje površi tela sa njihovom bojom i osvetljenošću koja varira zavisno od ugla, udaljenosti i intenziteta izvora svetlosti zaista više ne predstavlja ništa posebno.

U međuvremenu su se radnih stanica na PC tržište slili i novi paketi sa mogućnostima fotorealistične materijalizacije, teksturama, animacijom i ostalim neophodnim osobinama iz arsenala realističkog predstavljanje objekata. Jeden među njima osmišljen tako da se lako uklopi u najveći deo popularnih CAD programa je „Pixarov" *RenderMan*. Glavna odlika novog *AutoShada 2*, porad bilnih sopstvenih poodlišanja upravo i jeste ugradnja *RenderMan* paketa.

U ovom tastu pogledaćemo mogućnosti samog *AutoShade* koji je u novoj varziji dobio niz zanimljivih unapređenja. U idućem broju naša tema biće *Autodesk Renderman*.

AutoShada 2 386 se *RenderMan*-om dolazi na sedam HD diskala. Pet od njih pripadaju *AutoShade*-u sa *RenderMan*-om a dva *AutoFlix*-u. Instalacija je automatizovana praka za Autodeskove proizvoda već klsaičnog INSTALL programa firme „Knowledge Dynamics". Dokumentacija obuhvata četiri priručnika. „User Guide" (oko 300 strana) je glevni priručnik za korišćenja *AutoShade 2* i *RenderMan*-a a tu su „Installation Guide" (oko 100 atrana), „Tutorial" (koga, greškom nismo odbili) kao i „AutoFlix User Guide". „AutoFlix" je dodatni program koji omogućava ganarisanja *AutoShade* animacija na žalosi samo u EGA razolucji i bojama. To ga praktično čini neupotrabljivim – mnogo je odlja uz korišćanja skriptove, nepraviti animaciju kao niz TIFF alika i preoblikovati ja u FLi u *Animator Pro*-u.

ZA NAJTEŽA ORUDA

Za razliku od ranijih teslova „Autodeskovih" pakala kod *AutoShade* ćemo napraviti jeden mali izuzetak u radosleđu prikaza. Neima dis kusija vezana za potrabam hardver dolazi pre opise mogućnosti paketa. Razlog je jedan ali vredan hardverski zahtevi za *AutoShade 2 386* sa *RenderMan*-om su lakvi da za ona koji ne nameravaju da se oprama baram ovim minimumom koji navodimo, dalja razmišljanja o *RenderMan*-u nama mnogo smisla.

Ako žalimo da konstimo samo *AutoShade 386* bez *RenderMan*-a za sam start dovoljno nam je 4 Mb RAM-a i 100 Mb disk. Prosečni 33 MHz 386 sa koprocesoprom i prosečnom SVGA kartom sa ADi P386 drajverom i 800×600 razolucijom u 256 odja predstavlja najnižu granicu od koja se može poći u upotrebi *AutoCAD 11* + *AutoShade 2* paketa. I to samo za proetije modele. Sva iole sloenije biće pravo mučenje na takvom računaru.

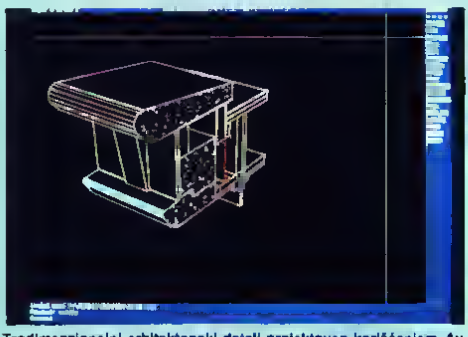
Ako radite sa složenim modelima, i uz to želite fotorealistična materijalizacija *Render Man*-a, MINIMUM za vas je 33 MHz 486 sa velikim sekundardnim kašom. 16 Mb RAM-a i 340 Mb tvrdim diskom. Grafika mora biti barem 15-bitna (32 768 boja istovremano od 16 7 miliona – nove HiColor SVGA karte je omogućavaju). „Waitak" 4167 koprocesor ili novi „Intalov" 66/33 MHz P 24 Turbo486 će dodatno ubrzati stvar (ovaj drugi malo više, a i u svim poslovima). Sa ovakvom konfiguracijom već možete početi da razmišljata i o genarisanju TGA animacija u punom koloru (16.7 miliona odja istovremeno) S-VHS ili, zašto ne, HDTV kvalitata.

A ako žalita da započeto razmišljanje i završite i zalata i krenete u genarisanje animacionih sakvanci, 50 MHz ili brži EISA 486 dok ne stigne 586 sa 32 do 64 Mb prepletenog RAM-a. 1 Gb brzim SCSI-2 diskom, izbrisivim optičkim diskom za arhiviranje animacija kao i inteligentnom grafikom 1024×768 u 16 7 miliona odja i video interfejsom, obaveza je za ozbiljan rad. Istina je da će *3D Studio* zahtevati nešto manja, ali kvalitat renderinga u njemu je, iz razloga koji će biti objašnjani, slabiji nego u *RenderMan*-u.

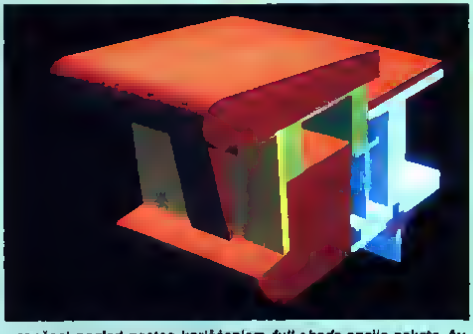
AutoShade 2 sa *RenderMan*-om neposredno podržava relativno mali broj grafičkih adaptera i izlaznih uređaja. Sam *AutoShade* deo (baz *RenderMan*-a) podržava ADi i ADI P386 drajvera za ekran, ulaz i izlaz, dok je najveći deo ugrađenih drajvera neupotrabljiv za ozbiljan rad – šta reći o ugrađenoj podrši za Herculas ili CGA grafiku? Sreća je što svaka iole ozbiljnija kartica danas ima ADI P386 drajver za *AutoCAD 11*. *AutoShade* i *3D Studio*. Gotov *AutoShade* rendaring može biti sačuvan kao PostScript ili RND fajl. Kao ulazni uređaji neposredno su podržani „Microsoft" miš i „Summagraphics" labia. *RenderMan* deo dodatno podržava „Truevision" TARGA i VISTA kartice, kolor PostScript kao i nekoliko izlaznih formata slika bliže opisanih u delu o *RenderMan*-u. S druga strane, *RenderMan* se uopšta na može pokrenuti sa običnim *real* mode ADI drajvenma. Morata imati ADI P366 drajvera za vašu grafičku karticu uz naravno, barem 32.768 boja istovremeno dostupnih iz palela od 16 miliona de bi stvar radila kako treba.

IZ AUTOCAD-a...

Proces genarisanje rendannga *AutoCAD* modela u *AutoShade* sastoji se iz dve faza. Prva faza, koja se obavlja u *AutoCAD*-u je postavljanje kamera i izvora svatiosli kao i zadavanje svojslava alemanata, odnosno površina



Trodimanzionalni arhitektonski detalj projektovan korišćenjem *AutoCAD* sistka *AEC UK 2 1*...



..senčeni pogled nastao korišćenjem *full shade* opcije paketa *AutoShade 2*.

modela. Polom se generiše *filmroll* (FLM) fajl koji predstavlja ulaz u *AutoShade*. Slično samo našlo složenji je i pul generisanja FLM fajla za „Autodesk *RenderMan*. Pogledajmo ovu fazu pobliže.

Posle dosta utrošenog vremena u projektovanja i premorenih očiju (osim eko imate monitor sa smanjenim zračenjem i visokom frekvencijom osvežavanja). *AutoCAD 11* model je završen. All, gomila linija na ekranu na dočaravanom objekat realistično. A kako bi to tek izgledalo investitoru.

Naravno naredba DVIEW za perspektivno sagledavanje objekta primenjena u ugrađenu *AutoCAD* naredbu SHADE daje neki ograničeni stepen prikaza „kako bi stvar mogla izgledati". Nju pre dole navedenih koraka i nije loše primeniti za probu i otkrivanje dobrih pogleda.

Prvi korak je biranje opcije ASHADE iz padajućeg **Utility** menija. Njom se učitava odgovarajući ASHADE LSP program, kojim nam postaju dostupna naredba za postavljanje svetala, kamera itd. Najpre treba postaviti svetla startovanjem naredba LIGHTS.

Pri postavljanju svetala treba voditi računa o tome da sa pozicije kada se uđe u *AutoShade*, ne mogu dodavati nova svetla kao ni menjati položaj i tip postojećih. S druge strane, slobodno možete definisati koliko hoćete svetala jer se bilo koja od njih posla u *AutoShade* može isključiti. Svako od svetala treba da dobije svoje ime, posla čega mu se odredjuje tip (tačkasto (svetlost u svim pravcima) ili usmereno (paralelni zraci – reflektor). Potom svetlu određuje željeni intenzitet i smestite ga u prostor modela. Ne lom mestu pojavljuje se „ikonica" svetla, koja je u *AutoCAD* DWG formatu ineče blok.

Posle svetala na redu su posmatrači – kamere. Postupak za definisanje kamera naredbom CAMERA sličan je onome za definisanje svetala. I ovim što treba odrediti vrednosti za druga parametra. Pored imena i koordinata kamera treba definisati koordinate ciljne tačke. Ostali parametri (promena pozicije sočiva kamera, rotacija oko ciljne tačke) definišu se u *AutoShade*.

Ako ste u perspektivnom modu rada *AutoCAD*, naredbama VLIGHT i VCAMERA možete postaviti svetlo odnosno kameru automatski na trenutno mesto posmatrača.

Novosti *AutoShade 2* u odnosu na ranije verzije je naredbe FINISH za definisanje optičkih osobina površina različitih elemenata modela. Na primer neke površine ili objekti mogu biti sjajniji od drugih. Svakoj od 256 *AutoCAD* boja možete oddelili određene vrednosti refleksije svetlosti i ponda će se te vrednosti automatski odnositi na sve elemente iste boje u modelu.

Kada su definisane kamera, svetla i svojstva površina na redu je previjenja scena. Svaka scena sadrži po jednu kameru i proizvoljan broj svetala. Kada smo definisali onoliko scena koliko nam treba naredbom FILMROLL generišemo sa ulazni fajl za *AutoShade* sa ekstenzijom .FLM (*filmroll* jednako može da posluži i kao ulaz u *3D Studio*). Potom izlazimo iz *AutoCAD*-a.

...U AUTOSHADE

AutoShade se startuje naredbom SHADE. Ukoliko je *filmroll* fajl s kojim želimo da radimo u istom direktorijumu kao i *AutoShade* sa SHADE <ime fajla (bez FLM)> možemo startovati program tako da automatski učita željeni *filmroll* i bude odmah spreman za rad s njim. Uz navodenje puta isto možemo uraditi i ako je *filmroll* u drugom direktorijumu.

Po startovanju programa pojavljuje se glavni ekran *AutoShade* koji je podeljen u tri dela najviši red sa padajućim menijima, najniži statusni red i preostali radni deo na kojem se pored ravanskih i žičanih prikaza modela, pojavljuju i dijalog prozori kod izvršavanja većine naredbi. Naredbe se startuju iz padajućih menija mada se najveći deo njih ako baš nemate miša ili grafičku tablu može pokrenuti i sa tastature. Padajući meni *AutoShade* ima četiri podmenija: **Display Settings RenderMan** (ukoliko ste kupili 386 varziju sa *RenderManom*) i **File**.

Ako prilikom startovanja nismo navali ime *filmroll*-a s kojim želimo da radimo, to nam je prvi korak. Polom biramo scenu iz *filmroll*-a. Naredbe iz **Display** padajućeg menija će nam omogućili da pogledamo model na više načina:

Plan View – standardni ravanskr pogled odozgo na osnovu modela sa učitanim svetlima kamerom i njenim vidnim poljem.

Wireframe – pogled posmatrača (kamera) na scenu sa klesičnim vektorskim (žičanim) prikazom modela – kao *AutoCAD*. Zbog brzine dobar je za proveru podešenosti položaja i parametera kamera pre renderinga.

Quick Shade – brzo kvalitetno senčenje 3D modela. Pogoden za računare sa velikim RAM om (8 Mb i više) jer proizvodi rastersku sliku umesto kodiranog niza poligona kao druga dva načina senčenja. Rasterska slika se brže generiše i prikazuje nego poligonalna ali se kod **Quick Shade** lako može desiti da na dodiru ivica dve različito bojene površine naslane linija neke lažne srednje boje – „edge artifact". Svoju brzinsku prednost u odnosu na druga dva načina **Quick Shade** duguje brzom „Z-Buffer" algoritmu za senčenje sličnom onome kod *AutoCAD* SHADE naredbe.

Fast Shade – računari sa manje memorije ovaj način će koristiti kao proveru osenčenosti površina pre **Full Shade**. **Fast Shade** žrtvuje preciznost radi brzine lako da po pravilu greši prilikom prikazivanja nekih površi.

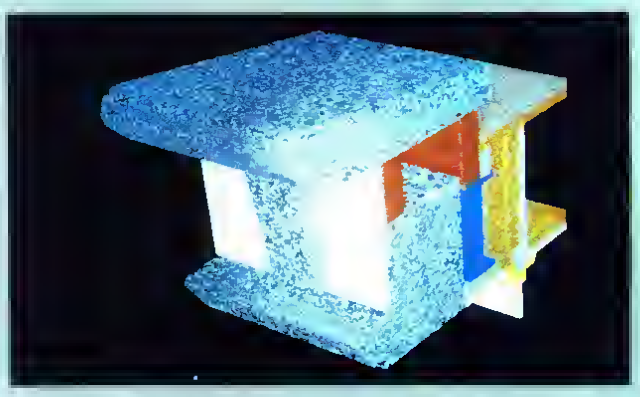
Full Shade – najkvalitetniji, ali i najsporiji način prikaza. Eliminiše „edge artifacts" i omogućava zaobljeno prikazivanje zaknvljenih površine prikazanih sa velikim brojem poligona.

Hard Copy opcija **Display** menija omogućava prenos gotovog renderinga na papir (jedini zaista prihvatljiv način je PostScript). U ovom meniju su još i opcija za reprodukciju jednog ili više gotovih renderinga pravljenja *AutoCAD* slajdova ili DXB (**Drawing eXchange Binary**) fajlova koje *AutoCAD* može unsti kao crteže. Bilan deo **Display** menija je i naredba za snimanje renderinga u RND, TGA ili TIFF formatu.

Settinga meni sadrži naredbe za podešavanje parametara kamera svetala senčenja svojstava površina, fino podešavanje pogleda na scenu kao i opciju stereoskopskog pogleda na scenu. Sva vrednosti koje su već bile dalinisane u *AutoCAD* ovde se mogu prepodesiti po potrebi.

File meni sadrži naredba za učitavanje *filmroll*-a snimanje svih trenutnih vrednosti parametara scena u poseban skript fajl, učitavanja drugog skripta se diska informaciju o paketu i serijskom broju, crtanja Mandelbrot sela i naravno QUIT.

Skriptovi su jedna od mocnih odlika *AutoShade*. Držeći se priručnika, vi bez ulaske u *AutoShade* možete napisati u vidu običnog tekst fajla skript kojim će ta na primer automatski učitati željeni *filmroll* i scenu i zatim pomeranjem ili rotiranjem kamere oko ciljna tačke u nekim koracima napraviti animaciju od željenog broja slika u nekom od formata koje *AutoShade* podržava. Te slika posle možete montirati u jedinstven FLC animacioni fajl u *Animator Pro*-u i tamo ih i dodatno obradivati.



I konačni, rendarisani pogled, nastao korišćenjem Autodesk *Renderman*-a

JOŠ NE VREDI MNOGO

AutoShade je u svojoj drugoj verzji doživeo više bitnih unapredanje veću brzinu i fleksibilnost. Medutim bez *RenderMana*-a *AutoShade* ni sa dodatkom 2 na vredi mnogo za pravu profesionalnu upotrebu. Kao što ćemo u idućem broju videti „Autodesk" *RenderMan*, koji je ipak priča za sebe, donosi jedan sasvim novi kvalitet prezentacije na personalcima (ako se personalcam može nazvati onaj poslednji predlog iz dela o konfiguraciji). Pored *RenderMan*-a u idućem broju biće opisan i *AutoFlix* animacioni program.

Korisna adresa

OSA racunarski Inzenjering
11130 Beograd, Mice Stojkovića 8
Tel. 011/557-592, Fax. 559 845

DALEKO OD PERFEKTNOG

Windows verzija čuvenog paketa za obradu teksta _WordPerfect_ pojavila se na tržištu tek krajem prošle godine. Program donosi nekoliko uzbudljivih koncepcijskih noviteta ali, nimalo u skladu sa renomeom proizvođača, u krajnje sumnjivoj realizaciji, koja je dovela program na ivicu upotrebljivosti.

Dejan Ristanović

Korporacija *WordPerfect* već duže vreme gradi svoju reputaciju i svoj gotovo neverovatni tržišni uspeh na nastojanju da bude za korak ispred konkurencije – prvi su na tržište profesionalnih tekst procesora za PC uveli (i delotvorno obradili) pisanje cirkularnih pisama, višestruko automatsko numerisanje objekata, „pomešali" tekst procesor i DTP paket podržavajući najrazličitije laserske štampače u verziji 5.0, uveli detaljan internacionalni set karaktera, za sam mnogo pre najvećeg konkurenta (*Microsoft Word*) podržali rad sa formulama i tabelama. Ne treba, naravno, zaboraviti ni verzije *WordPerfect*-a za druge operativne sisteme kao što je VMS. Ipak, *Windows*-u 3.0, operativnom sistemu koji je još 1990. godine privukao pažnju svih autora softvera, su se prilagođavali malo duže nego što je uobičajeno – dok je *Microsoft* izbacio par verzija programa *Word for Windows*, korporacija *WordPerfect* je čekala i tek krajem 1991. lansirala dugo očekivan *WordPerfect for Windows*. Da vidimo koliko su pri tome bili uspešni.

WordPerfect for Windows (korisnici ga često skraćeno zovu WP4WIN, pri čemu se broj 4 ne čita kao *four* nego kao *for*) koji ovde prikazujemo datiran je 4. novembra 1991. i nosi oznaku 5.1. To bi trebalo da asocira na jednako označenu verziju „običnog" *WordPerfect*-a (prikazali smo je još u „Računarima 60") što na prvi pogled uopšte nije očigledno – kada startujete *WordPerfect for Windows* videćete ekran koji ni najmanje ne liči na dobro poznatu DOS varijantu! Pa ipak, čim se ozbiljnije udubite u program, sličnosti izlaze na videlo – iste su to opcije ali drugačije „upakovane". *WordPerfect for Windows*, istini za volju, ima i određene novitete u odnosu na svog DOS prethodnika ali se, sve u svemu, može reći da je korporacija *WordPerfect* zaista pošteno dodelila oznaku verzije – u pitanju je zaista *WordPerfect 5.1* solidno prilagođen *Windows* okruženju.

POČETAK (NI)JE TEŽAK

Pre nego što pređemo na karakteristike, pozabavimo se hardverskim zahtevima i instalacijom. Za razliku od ranijih prikaza, o hardverskim zahtevima gotovo da nema potrebe pisati – da bi *WordPerfect for Windows* radio na vašem računaru mora da radi *Windows 3.0* i to u standardnom ili 386 modu. Nema, dakle, nikakvih ograničenja po pitanju grafičke kartice, štampača, memorije, miša... ako ste uspešno instalirali *Windows*, ne bi trebalo da imate problema ni sa ovom verzijom *WordPerfect*-a.

Onima koji tek planiraju da (možda ovim povodom) instaliraju *Windows* reći ćemo da im treba računar zasnovan na mikroprocesoru 80286, 80386 ili 80486 (zbog brzine je preporučljiv drugi ili treći) sa 2, 4 ili više megabajta memorije, *Hercules* ili VGA karticom (mi smo testirali program na SuperVGA grafičkoj kartici, uglavnom rezoluciju 800×600) i odgovarajućim monitorom. Za razliku od nekadašnjih verzija *WordPerfect*-a (4.1, 4.2, 5.0) koje nisu znale šta znači reč miš, WP4WIN zahteva ovaj dodatak – možda biste i bez njega mogli raditi, ali biste propustili neke od najvažnijih noviteta.

Što se štampača tiče, izbor je veoma širok. Možete se, pre svega, osloniti na bilo koji matrični, termički ili laserski štampač koji uspešno komunicira sa operativnim sistemom *Windows*. Mnogi korisnici će, među tim, ignorisati ovu mogućnost i instalirati jedan od tradicionalno brojnih drajvera koje isporučuje korporacija *WordPerfect*, iz razloga o kojima ćemo za trenutak govoriti. U bilo kom slučaju, štampač će (ako, naravno, ima grafičke mogućnosti) podržati sve opcije *WordPerfect*-a, uključujući tu i strane znake, formule, tabele, slike... Kvalitet izlaznog dokumenta prirodno zavisi od štampača, ali bi raspored objekata na papiru trebalo da bude isti na 9 pinova i na laseru.

Obzirom da *WordPerfect for Windows* radi na AT (i moćnijim) računarima, proizvođači su konačno odustali od isporuke na 360K disketama. Sva sreća – umesto 24 diskete od 360K (koliko bi kompletna instalacija svakako zahtevala) dobijate „samo" 6 disketa od po 1.44 megabajta. Instalacija počinje jednostavnim kucanjem komande INSTALL, posle čega treba odgovoriti na par pitanja o direktorijumima – *WordPerfect for Windows* se rasprostire u nekoliko direktorijuma od kojih je jedan ekskluzivno njegov, drugi namenjen povezivanju sa drugim *Windows* aplikacijama, a poddirektorijumi sadrže makroe, slike, primere i druge dodatke. Ako instalirate sve što se nudi, *WordPerfect for Windows* će zauzeti nešto manje od 10 megabajta prostora na disku, što znači da se radi o pravom gigantu. Ako odustanete od nekih elemenata i odreknete se rečnika za kontrolu spelovanja, rečnika sinonima, dinamičke veze sa drugim programima i tome slično, ostaće vam nekih 6 megabajta, što je i dalje ogromno, naročito kada se uzme u obzir da, ukoliko želite raznovrstan ispis na laserskom štampaču, morate odvojiti dalje megabajte za fontove.

DVE VAŽNE ODLUKE

Prilikom instalacije programa *WordPerfect for Windows* morate da donesete dve veoma važne odluke. Treba se, pre svega, opredeliti za metod komuniciranja sa štampačem – preko *Windows*-a ili direktno. U nekom idealnom svetu ne bismo mnogo razmišljali pre nego što se opredelimo za prvu varijantu – na kraju krajeva, čitava ideja *Windows*-a je da jednom instalirate grafičku karticu, štampač i drugu specifičnu periferijsku opremu i da docnije svi *Windows* kompatibilni programi rade bez muka sa podešavanjem parametara, portova i drugih stvari. Probali smo ovu varijantu i možemo da kažemo da je radila. Ipak, nije radila baš savršeno. Pre svega, komunikacija je dosta spora, naročito kada se pomene reč grafika (a šta će nam *WordPerfect for Windows* ako ne za igre sa grafikom?). Drugo, pojavio se problem instaliranja brojnih raznovrsnih fontova sa YU slovima, koji su bazično namenjeni programu *Ventura Publisher* ali smo ih do sada uspešno koristili i sa *WordPerfect*-om – zašto da držimo na disku dva seta fontova (svaki po 7-8 megabajta) kad ova dva programa mogu da koriste jedan isti? Presudilo je to što smo u dokumentaciji pročitali da *WordPerfect for Windows* koristi potpuno iste drajvere za štampač kao i *WordPerfect 5.1*, pa smo se odlučili za takvu varijantu. U početku smo i sa njom imali nekih problema (WP4WIN je „pucao" kada smo mu rekli da uzme drajver iz starog *WordPerfect* direktorijuma, kada smo mu ga ručno prekopirali, sve je bilo u redu), ali su se, čim je proradila, na laseru pojavile stranice teksta sa poznatim fontovima i potrebnim YU slovima. Ukratko, ukoliko ste već koristili *WordPerfect 5.1* i povezali ga sa štampačem na pravi način, brže ćete obaviti posao ako se oslonite na stari drajver. Ako vam je ovo prvi kontakt sa *WordPerfect*-om i ako ste propisno instalirali *Windows* i radili sa njegovim fontovima, verovatno je najbolje da se opredelite za osnovnu koncepciju komunikacije preko operativnog sistema. I jedna i druga varijanta bi na kraju trebale da proizvedu isti rezultat.

Druga odluka se odnosi na sistem komandi. *WordPerfect* je na tržištu još od dana kada su standardni metodi komunikacije sa PC računarom bili prilično nerazvijeni, i kada je autor svakog programa izmišljao korisnički interfejs prema svojim idejama. *WordPerfect*-ov korisnički interfejs je uvek bio pomalo „čudan", a funkcije su raspoređene po tasterima na način koji je malo koji drugi proizvođač softvera prihvatio. Taj raspored se, uz to, menjao iz verzije u verziju *WordPerfect*-a pošto je trebalo dodavati nove funkcije ali i sistematizovati postojeće – nije bilo mnogo veze sa „tuđim" korisničkim interfejsom, ali su to korisnici ovog tekst procesora nekako prihvatali. Prelazak na *Windows*, međutim, ne znači samo standardizaciju grafičkih kartica i štampača – treba usvojiti i standardni sistem komandi u kome se HELP dobija sa F1 a ne sa F3, gde se tekst snima sa F3 a ne sa F10, štampa sa F5 a ne sa Shift F7 i tako dalje. Kada, sa druge strane, pišete tekst, poželjno je da razmišljate o ono



mu što ćete napisati, a ne o tome gde je koja komanda. Je se na promene ovoga tipa prilično teško navići – funkcijski tasteri još i nekako, ali odustajanje od WordPerfect-ova prakse da se taster Home koristi kao nekakav dodatni Shift (na početak reda se, na primer, dolazi pritiskom na Home i levu strelicu) zahteva priličan napor korisnika. Zato je WordPerfect obezbedio opciono korišćenje starih komandi, naravno ukoliko se pri instalaciji izabere odgovarajuća opcija (izbor takođe nije konačan – ako zaključite da vam komande ne odgovaraju, uvek možete izabrati onu drugu grupu). Kada se opredelite za jednu od skupina komandi, WordPerfect će vas osloboditi obaveze da razmišljate o onoj drugoj – u svim HELP-ovima i menijima pominju se samo komande koje ste izabrali. Pa ipak, ova dva skupa nisu baš ravnopravna – u čitavoj dokumentaciji pominju se samo *Windows* kompatibilne komande. To bi moglo da znači da *WordPerfect* prihvata poslovicu „ako ne možeš da ih pobediš, ti im se pridruži" i slaganje sa prihvaćenim standardima. Ukoliko vam je ovo prvi kontakt sa *WordPerfect*-om, svakako bismo vam savetovali da se opredelite za *Windows* kompatibilne komande, jer ćete tako sebi olakšati korišćenje uputstva, tako i rad sa drugim programima. Ukoliko ste iskusni korisnik DOS verzija *WordPerfect*-a, uz sve što možete je da pokušate – izaberite *Windows* komande i videćete da im se možete prilagoditi. Ako ste za izbor možete da promenite, ali treba znati da ćete jednom svakako morati da menjate navike jer se protiv standarda ne može večito boriti – možda će biti lakše ako to „obavite" što pre.

KONCEPCIJSKI ZAOKRET

Promena sistema komandi nikako nije jedini koncepcijski novitet koji je *WordPerfect for Windows* doneo. Još kada smo u svoje vreme predstavljali *WordPerfect* čitaocima, rekli smo da on počiva na četiri osnovna principa koja smo nazvali „Koncept čistog ekrana", „Koncept skrivenih kodova", „Koncept neimenovanog dokumenta" i „Rad sa funkcijskim tasterima". Od ta četiri „stožera" ostao je samo jedan ipo, što moramo nazvati krupnim koncepcijskim zaokretom – zapravo, do sada nismo videli program koji bi, sačuvavši iste ili slične funkcije, **toliko** promenio svoj grafički izgled i korisnički interfejs kao *WordPerfect for Windows*!

Jedino što je potpuno ostalo na snazi jeste koncept neimenovanog dokumenta – kada počinjete da radite sa dokumentom, ne morate specificirati njegovo ime, što znači da tek pri snimanju morate da odlučite kako će se sve zvati, možete čak snimiti deo dokumenta pod drugim imenom. Delimično je na snazi koncept skrivenih kodova – svaka od funkcija (masna slova, kurziv, fusnote, grafički okviri itd) se upisuje u tekst kao skriven kod koga u normalnom modu na ekranu ali koji može da se provera pregleda pa i izbriše u posebnom *Reveal codes* modu u koji se ulazi pritiskom na F11 ili Alt F3. *Reveal codes* mod je zadržan, ali kodovi više nisu sasvim skriveni pošto je WYSIWYG (*what you see is what you get*) filozofija napredovala, pa se na ekranu vidi veličina slova, kurziv i slično.

Od koncepta čistog ekrana *WordPerfect* je definitivno odustao – umesto da ceo ekran bude posvećen dokumentu, *WordPerfect for Windows* (što se da videti sa slike) ima uobičajene linije za skrolovanje desno i dole, meni u vrhu, status liniju na dnu a po potrebi se pojavljuju lenjir (*ruler*), čitava dodatna tastatura, meniji padaju na tekst i tome slično. Obzirom da je rezolucija znatno veća na ekran i pored njih dokumentu ostaje podosta teksta, zavisno od izabranog fonta, mnogo više nego kod DOS verzije. Pa ipak, se sitnim slovima se teže radi, tako da je ostavljena mogućnost uklanjanja nekih od elemenata, pri čemu meniji (za razliku od DOS verzije) moraju ostati na ekranu.

Postojanje menija znači da se *WordPerfect* bar delimično odrekao svog „zaštitnog znaka" – rada sa funkcijskim tasterima. Ovo odricanje je počelo još od *WordPerfect*-a 5.1 za DOS (koji je dobio detaljne menije), ali je tada osnovni način komuniciranja i dalje bio zasnovan na funkcijskim tasterima (što se lako uočavalo u dokumentaciji) dok su meniji više bili namenjeni snalaženju onih koji tek počinju. Sada je obrnuto – i dalje se možete opsluživati funkcijskim tasterima, ali je kroz čitavu dokumentaciju preporučen rad sa menijima koji su, uostalom, obavezni element jedne *Windows* aplikacije koja drži do sebe.

U osnovnom režimu rada na raspolaganju su meniji **File** (učitavanje, snimanje i štampanje tekstova, podešavanje štampača i slične operacije), **Edit** (premeštanje segmenata teksta, pretraživanje, promena velikih slova u mala, povezivanja sa drugim aplikacijama itd), **View** (određivanje elemenata koji se prikazuju na ekranu), **Layout** (definisanje oblika strane, margina i uravnanja, zadavanje i podešavanja fusnota), **Tools** (program za kontrolu spelovanja, rečnik sinonima, prebrojavanje reči, obeležavanje poglavlja, generisanje sadržaja i indeksa, poređenje dokumenata itd), **Font** (određivanja veličine i tipa slova kao i izbor naglašavanja segmenata dokumenta), **Graphics** (rad sa grafikom i matematičkim formulama), **Macro** (definisanja i izvršavanja makro naredbi), **Window** (izbor radnog prozora) i **Help**. Obzirom da meniji imaju pod menija, broj raspoloživih funkcija je prilično veliki (mi smo, uz svu opasnost da nešto propustimo, nabrojali preko 70) pa ih zaista nije lako rasporediti po funkcijskim tasterima. Nije, kažemo, lako ali se *WordPerfect* potrudio – (skoro) sve funkcije mogu da se realizuju i sa tastature ali se, uz funkcijske tastere kombinovana sa Alt, Shift i Ctrl sada koriste i kombinacija Ctrl-Shift-Fn i Alt-Shift+Fn gde je Fn neki od tastera F1, F2, F3, ... F12. Iako je primetan pokušaj da stvari budu logične (Alt–Shift i Ctrl–Shift kombinacija se obično koristi za funkciju koja je na neki način suprotna onoj funkciji koja se dobija bez Alt odnosno bez Ctrl – pretraživanje unapred i unazad, prelazak na sledeći odnosno prethodni dokument i tome slično), čini nam se da funkcija jednostavno ima previše da bi od korisnika očekivalo da ih zapamti, pa čak i da ih traži po kartonu koji *WordPerfect* daje uz program. Pa ipak, ukoliko ste nekada u situaciji da više puta ponavljate neku operaciju (što se kod obrade teksta ne dešava tako retko), isplatiće se da koristite komandu umesto da šetate kroz par menija. U normalnom radu je daleko lakše osloniti se na miša, naročito u kombinaciji sa novitetom o kome ćemo za trenutak govoriti.

TASTATURA NA EKRANU

Svi znamo koliko su retki noviteti u domenu korisničkog interfejsa – *Xerox*, *Apple*, *Microsoft* se i dan danas sude (a nije neverovatno da će spor potrajati i do sledeće decenije) o tome ko je izmislio raznorazna skačuća i padajuća menije, a najsitnija razlika često biva osnov za zaštitu patenta. *WordPerfect* je u domen korisničkog interfejsa uveo nešto sasvim novo, što nas je, moramo priznati, vrlo prijatno iznenadilo. Znajući koliko takvi noviteti mogu da budu značajni, *WordPerfect* je zaštitio svoja patentna prava tako da već na prvoj strani dokumentacije piše da su reči *WordPerfect* i *Button Bar* zaštićena oznake (*registered trademark*) korporacije *WordPerfect*. Pošto znamo šta je *WordPerfect*, ostaje da upoznamo apsolutni novitet zvani *Button bar* ili, da pokušamo sa slobodnim prevodom, tasterski meni.

Dobra strana menija je, znamo, što je očigledan i pristupačan korisniku. Loša strana je što, ako postoji 2 ili 3 nivoa menija, dolazak do neke opcije malo predugo traje (naročito ako ga često obavljate), a ponekad se i nije lako setiti u kom meniju ili podmeniju se krije neka funkcija. Meni i dalje ostaje jedini način da se izađe na kraj sa ogromnim brojem funkcija, ali u realnom radu se svega nekoliko funkcija koristi jako često, dok ostale i nisu toliko bitne. Konkretno, neprekidno nam je potrebno učitavanje, snimanje i štampanje dokumenta, prenošenje blokova, izbor naglašavanja teksta (bold, kurziv itd) i možda pretraživanje. *WordPerfect* je došao na ideju da funkcijama tog tipa dodeli posebne ikone i da ih, u vidu neke vrste tastera, postavi u jedan red ekrana. Pošto je ovaj tasterski meni uključen, pozicioniranjem kurzora na neki od „tastera" i pritiskom na levi senzor miša aktivirate željenu funkciju. Umesto, dakle, da pozivate meni **Edit** i tražite u njemu funkciju *Paste*, jednostavno ćete kliknuti na ikonu *Paste* u tasterskom meniju. Strašno jednostavno i veoma efektno.

Pravi efekat dobijate tek kada primetite da se tasterski meni može konfigurisati – najzad, funkcije koje su potrebne jednom korisniku ne moraju odgovarati ukusu nekog drugog! Sam *WordPerfect* je dao osnovna tasterska menije, a korisnik ih može ispravljati izborom stavke *Button Bar Setup* iz **View** menija. Samo ispravljanje je jako jednostavno – aktivira se neki meni i izabere neka njegova stavka, automatski se pojavi njena ikona (moramo reći da su ikone veoma duhovito i inventivno izabrane, naprosto smo morali da šetamo kroz menije da bismo videli kako izgledaju i grafički opisi onih funkcija koje nismo ni planirali da uključimo u *Button Bar*) koju onda pozicionirate tamo gde želite. Postojeće tastere možete da razmestite na neki drugi način ili čak uklonite iz menija, a zatim ispravljeni tasterski meni snimate na disk. Vredi reći da *WordPerfect for Windows* ne nudi samo jedan tasterski meni, nego sadržaj menija zavisi od okolnosti (*context sensitive*). Ukoliko, na primer, pređete u mod za unošenje formula ili tabela, pojavljuju se odvojeni tasterski meniji sa funkcijama koje su u tim prilikama potrebne. Svaki od ovih menija se može ispravljati, a onda se svi zajedno (ili samo neki od njih) mogu snimiti na disk pod izabranim imenom. Osim standardnih funkcija samog *WordPerfect*-a, u tasterski meni mogu da se uključuju i pozivi makro naredbi, čime se proizvoda čitave biblioteka tasterskih menija pogodnih za pojedine vrste posla, ne sumnjamo da će se time baviti i nezavisne softverske firme.

Što se nas tiče, jedina izmena koju smo našli potrebno da izvršimo bila je dodavanje funkcija *Undo* (poništavanje prethodne operacije, mnogo pogodnije od u ranijim verzijama *WordPerfect*-a popularnog *Cancel*) i *Search back* u osnovni tasterski meni ... nam se da su svi ostali optimalno raspoređeni. Eksperimentisali smo i sa pozicijom samog tasterskog menija (možete ga postaviti iznad, ispod, levo ili desno od glavnog ekrana, nekim korisnicima Sezama se posebno dopalo da ga stave dole što zbilja avangardno izgleda), ali smo ga na kraju ostavili na početnoj poziciji, između „pravih" menija i teksta.

DA LI JE WYSIWYG

Kada smo već kod tasterskog menija, da pomenemo i lenjir (*ruler*) koji, dakako, nije pronalazak korporacije *WordPerfect* ali koji je ta korporacija do sada iz sve snage izbegavala, valjda zbog spomenutog koncepta čistog ekrana. Radi se o linij ... izdeljenoj na centimetre i milimetre (naravno, podrazumevano su inči, ali se to lako izmeni) po kojoj „šetaju" markeri margina, pozicije kolona, tabulatore... Sve to naravno može zadavati i u milimetrima posredstvom **Layout** menija, ali je najčešće sasvim dovoljno (i znatno brže i jednostavnije) mišem „povući" markere, a docnije eventualno vršiti fino podešavanje iz menija. Osim klasičnih funkcija, lenjir ima i svoj tasterski meni (koji se ne može editovati) preko koga se podešavaju font, uravnanja i razmak između redova. Posebni deo lenjira je posvećen tabelama.

LIČNA KARTA

Program WordPerfect for Windows
Verzija 5.1, 4. novembar 1991
Namena Tekst procesor
Sadržaj paketa 6 disketa 1.44 megabajta i dve knjige
Hardversko-softversko okruženje IBM AT/386 ili kompatibilan, 2 M RAM-a, hard disk, Windows 3.0 ili noviji
Zauzet prostor na disku oko 9 megabajta ne računajući fontova
Ulaz Tastatura, miš
Izlaz – ekran bilo koja Windows kompatibilna kartica
Dokumentacija WordPerfect User Manual 850 strana
WordPerfect WorkBook 280 strana

Proizvođač	WordPerfect Corp 1555 N. Technology Way OREM, UT 84057, U.S.A. Tel. (801) 225-5000 Fax (801) 222-5077
Predstavnik	Perpetuum d.o.o. Kučerina 5 41000 Zagreb Tel. (041) 317-020 Fax (041) 328-999

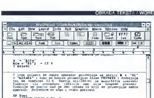
ZA I PROTIV

Hvalimo

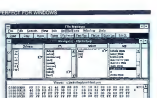
1. Ekranske tastere (button bar)
2. Editor formula
3. Rad sa više dokumenata
4. Ogroman set karaktera
5. Solidan WYSIWYG
6. Brojna sitna poboljšanja

Kritikujemo

1. Brojna bagova koji čine verziju neupotrebljivom
2. Loš izgled pojedinih stranih slova
3. Ukidanje ALTx makroa



Novost u korisničkom interfejsu: *Button bar*



Kretanja po disku i datotekama: *Navigator*

Već pomenuta fraza WYSIWYG iz dana u dan ima sve manje smisla. Sećamo se da se prvi put pojavila u prikazu nekih od starih verzija *WordStar*-a kada je označavala da se zamislite na ekranu vidi gde je kraj koje strane! Kako je tehnika napredovala, tako su se i zahtevi povećavali: da se vidi tačan prelom redova, pa da se vide kurzivna slova, zatim formula, uvećana i umanjena slova, fusnote... tako se i pojavila fraza WYSIRWYG (*what you see is REALLY what you get*). Zapravo, nećete mnogo pogrešiti ako za bilo koji današnji tekst procesor kažete da je WYSIWYG, ali isto tako nećete pogrešiti ni ako kažete da na tržištu uopšte nema pravog WYSIWYG procesora za PC.

Obzirom da se preselio u grafički mod *Windows*-a, novi *WordPerfect* bi po prirodi stvari trebalo da bude „više WYSIWYG" nego prethodna verzija. On to i jeste, ali i dalje ima stvari koje se na ekranu ne prikazuju. Da bi sve bilo još čudnije, *WordPerfect* je uveo čudnu koncepciju rada u tri moda.

Draft mod je namenjen „sirovom" unošenju teksta – radi se u neproporcionalnom fontu, što znači da su sva slova jednako široka, a pojedina isticanja (kurziv, masna slova itd) se označavaju različitim bojama, baš kao u DOS verziji *WordPerfect*-a. Mišljenja smo da ovaj mod ima smisla jedino na sporim mašinama (AT) koje ne mogu da „izdrže" osvežavanja realnog ekrana; na 80386 i 80486 računarima nema ga smisla aktivirati.

Standardni mod omogućava unos i ispravljanje teksta uz vernu simulaciju oblika i veličina slova (veća slova su zaista veća, *Times* se razlikuje od *Helvetice*, indeksi i izložioci se vide na ekranu) te izgleda redova i same strane (redovi su razmaknuti koliko treba, desna ivica je po potrebi uravnata, pozicije tabulatora odgovaraju realnim odnosima, vidi se prelazak sa strane na stranu, tabele su ukrašene svim potrebnim linijama, kolone su korektno formatirane i tome slično). Pa ipak, ni u ovom modu se ne vide slike (zamenjuje ih prazan pravougaonik odgovarajuće veličine), formule (što je prilično neprijatno), zaglavlja i potpis strana te fusnote i slični elementi. Da biste videli kako **stvarno** izgleda strana, morate da pređete u *preview* mod u kome možete posmatrati i susedne strane, uvećavati i umanjivati tekst, štampati i tome slično (naravno, sve ovo se obavlja uz pomoć posebnog lastarskog menija).

Ovakva koncepcija u praksi pokazuje određene dobre i loše strane. Dobre: što je rad izuzetno ubrzan, što pri većini operacija zaista i nije mnogo bitno kako strana **stvarno** izgleda – to se bezbroj puta promeni pre nego što dođemo do završne faze kada uopšte ima smisla ići u *preview* mod. Sa druge strane, prelazak u *preview* uzima određeno (ne baš neprijatno dugo, ali nikako zanemarljivo!) vreme, što nije zgodno ako se dosta eksperimentiše sa tekstom.

Ukratko, po WYSIWYG planu *WordPerfect for Windows* i dalje ulija „stignu" *Venturu* koja u svakom trenutku prikazuje kompletnu stranu sa sve fusnotama i slikama. Sa druge strane, korak unapred je učinjen jer se, pored svih pomenutih stvari, slike sada mogu slobodno pomerati mišem po strani umesto da se zahteva stalno zadavanje dimenzija. Najzad, uzmimo u obzir da *Ventura*, pored WYSIWYG prikaza, praktično ne omogućava rad sa tekstom, dok je *WordPerfect for Windows* pre svega tekst procesor pa tek onda DTP paket. Vrlo je moguće da performanse sadašnjih personalnih računara i široko prisutnih Super VGA kartica objektivno ne omogućavaju kompletan WYSIWYG rad... no uz malo strpljenja dočekaćemo 80586, nov operativni sistem i *WordPerfect 6.0* za njega, možda će tada izraz WYSIWYG još jednom dobiti novi smisao.

LAKO SA FORMULAMA

Pored ubedljivo prvog mesta na svetskoj listi najprodavanijih tekst procesora, *WordPerfect* uživa sve veću popularnost i u našim krajevima. Razlog zbog koga kod nas nije još popularniji je svakako u tome što je „prekasno" dobio mogućnost rada sa formulama – neki tekst procesori koji mu po mogućnostima ne mogu konkurisati ali koji su obezbeđivali unošenje i ispravljanje formula su tako stekli široku korisničku bazu koja se teško prebacuje na „nešto novo". Često se čulo i pitanje zašto je korporacija *WordPerfect* toliko čekala sa formulama i orijentisala se na razne „glupe" opcije za *mail merge*, generisanje unakrsnih referenci, formiranje sadržaja odnosno indeksa? Najverovatnije zato što su tekst procesori obično pravljeni ne za jugoslovensko nego za američko tržište i prema tome ciljaju na različitu grupu korisnika – dok su kod nas računari i dalje najvećim delom u rukama „tehničke inteligencije" kojoj su formule neprekidno potrebne, osnova američkog tržišta su ljudi različitih profesija koji kompjuter koriste isključivo kao alatku; njima formule ne znače baš ništa, dok će, recimo, mogućnost vođenja i uparivanja beleški o rezultatima predmeta predstavljati momentalnu uštedu u ogromnom broju kompanija koje poseduju pravnu službu. Strpljenje se najzad uvek isplati, pa smo tako i svi mi koji smo godinama ostali verni *WordPerfect*-u dočekali i do matematike – uz razne druge izmene *WordPerfect 5.1* je uveo rad sa formulama koji je u izvesnoj meri unapređen u *Windows* verziji.

Editor formula je i dalje uklopljen u postojeće operacije *WordPerfect*-a 5.0 za rad sa grafikom: po aktiviranju menija **Graphics** dobijamo priliku da kreiramo sliku, tabelu, ram sa tekstom, ram proizvoljnog tipa i od sada formulu (*Equation*). Kao i svaka druga „slika", formula može da se priveže za stranu, pasus ili slovo.

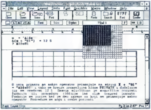
Formula vezana za stranu teško da ima smisla – ova je opcija ostavljena isključivo zbog univerzalne kompatibilnosti svih tipova ilustracija u *WordPerfect*-u; može se zamisliti da će neko na precizno određenu poziciju strane 5 staviti neku sliku, ali formula koja bi bila postavljena na neko mesto koje ne zavisi od okolnog teksta ne zvuči baš smisleno. Najčešće se koristi formula vezana za pasus – tekst završimo sa „u skladu sa prethodnim biće" a onda u sledećem redu navedemo formulu koja je obično centrirana između leve i desne margine. Po potrebi se ovakva formula može uokviriti, dopuniti potpisom, automatskom referencom i slično.

Formula vezana za karakter je tipično kratka formula koja se umeće u tekst, nešto u stilu „pošto je $p^3 = q^2 + 17$ onda je ...". Iako ova formula može biti složena koliko i svaka druga, nije preporučljivo da njena visina bude prevelika – *WordPerfect for Windows* će doduše u potrebnoj meri razmaknuti redove, ali takav tekst teško može zadovoljiti čak i najmanje estetske kriterijume. Mana *WordPerfect*-a je što podrazumevane (*default*) vrednosti kod formula vezanih za karakter nisu dobro izabrane – za svaku ovakvu formulu treba posebno zadati *Size: Auto Both* i *Vertical Position: Baseline* – ako u tekstu ima mnogo formula, ove dve operacije će vam začas dosaditi, pa ćete morati da se snalazite sa makro naredbama ili nekim drugim trikovima.

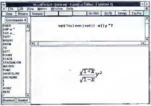
Pošto je formula kreirana, treba je i otkucati; za šta se koristi grafički editor sa slike: ekran je podeljen na prozor za unos formule, prozor za njen pregled i HELP prozor (levo) sa komandama editora. Za unos formule treba znati i pomalo matematike – kuca se neka vrsta bejzik ekvivalenta: **2^3** je običan 2^3, **sqrt** (2+3) uvodi simbol za kvadratni koren, **sqrt** {sqrt 4 over sqrt {5}} bi bio razlomak pod korenom i tome slično. Za početak treba zapamtiti da se umesto običnih zagrada izrazi uokviravaju vitičastim zagradama { i } i da se sa *Screen* (Ctrl F3) dobija izgled trenutno unesene formule uz, naravno, moguću poruku o sintaksnoj grešci *Incorrect format* – kontrola sintakse se uguralа i u tekst procesor!

Obzirom da je ovo prikaz a ne uputstvo za upotrebu *WordPerfect*-a, nećemo detaljnije opisivati sintaksu kojom se formule zadaju – reći ćemo samo da je na raspolaganju praktično sve što će vam **ikada** zatrebati. Osim nekih očekivanih stvari kao što su sum, prod, integral, sva moguća i nemoguća grčka slova, oznaka Laplasove transformacije, znaci „deljivo je", „nije deljivo", „paralelno" i „seče se", „ugao" itd, naći ćete i Plankovu konstantu, znak za „nikad ne postoji" ili čak „ne sledi niti je ekvivalentno" i svašta drugo. Sami znaci naravno nisu dovoljni – treba nekako iscrtati složenije šeme kao što su matrice, integrali i slično. I tu su se programeri korporacije *WordPerfect* pokazali: obezbedili su brojne koeficijente, sumu, integral, funkciju, matricu, kvadratni i n-ti koren i razlomak uz brojne „podopcije" koje omogućavaju slaganje proizvoljno složenih formula.

Nameće se, naravno, poređenje sa nečim što već postoji – sistem za slaganje formula ugrađen u paket *Ventura Professional 3.0*. Nije da nema sličnosti – formule se u osnovi unose na isti način, sintaksa je relativno slična, omogućen je *preview* formule... *WordPerfect for Windows* ipak ima jednu ogromnu prednost – *Ventura* slaže formule od slova tekućeg fonta i fonta simbola dok *WordPerfect* radi u grafičkom modu. U čemu je prednost? Jednostavno: ako *WordPerfect*-u treba slovo neke veličine, on će ga iscrtati. Ukoliko, sa druge strane, *Venturi* treba neki neobičan znak određene veličine, ona će ga potražiti u simbol fontu, a onda izabrati najpribližniju raspoloživu veličinu – ili ćete na disku imati raspakovane simbol fontove **svih** veličina ili ćete doći do publikacije koja će unakaraditi formulu! Nema ni poređenja između spiskova raspoloživih znakova – *WordPerfect* vam omogućava da izaberete bilo koji od 5000 znakova, dok je *Ventura* ograničena na svega par stotina najpotrebnijih simbola, da i ne računamo ograničenje od 16 fontova po *LaserJet II* strani o čemu *WordPerfect* ne mora da vodi računa kada kompajlira formule. Čini mi se, sve u svemu, da *WordPerfect for Windows* na polju matematičke tipografije prevazilazi *Venturu* za 10 dužina!



Interaktivno dimenzionisanje tabele



... i interaktivni rad sa formulama

Naravno, nije da nema i problema. Pre svega, *WordPerfect for Windows* u domenu formula nije sasvim WYSIWYG – formulu vidite u editoru, ali kada se vratite u tekst, vidite samo kockicu koja je zamenjuje; da biste videli kako izgleda čitava strana, morate prelaziti u *preview* mod. Osim toga, može se zamisliti i situacija u kojoj je neprijatno što *WordPerfect* samostalno komponuje formulu na osnovu sintakse koju ste zadali – ako vam se rezultat ne svđa, možete da dodajete blankove i druge separatore, ali su vam ruke u priličnoj meri vezane. Konstruktori *WordPerfect*-a su obezbedili brojne opcije za podešavanje (počevši od veličine fonta i međusobnih razmera (po veličini) susednih pisama do mnogo suptilnijih „sitnica" kao što su linije koje uokviravaju matrice, zagrade oko binomnih koeficijenata i slično) ali se uvek može zamisliti slučaj u kome nedostaje baš ono što vam je potrebno!

S TABELAMA NIŠTA NOVO

WordPerfect je od svojih početaka bio solidno opremljen oruđima za formiranje tabela – u tekst procesor je zapravo bio ugrađen kompletan programčić za unakrsna izračunavanja (*spreadsheet*) koji je mogao štampati rezultate svoga rada u tabelarnom obliku. Te tabele ipak nisu bile dovoljne – kada bi nam zatrebala išpartana kvadratna struktura, morali smo da se mučimo sa linijama, dvostrukim linijama i mnogim drugim prljavim trikovima i da se na kraju opredelimo za neki neproporcionalni font! Sada je opcija *Math* ukinuta (jedna od retkih stvari koje su postojale u DOS verziji *WordPerfect*-a i koje su „nestale" u *Windows* varijanti), ali je uvedeno nešto bolje: opcija *Tables* u **Layout** meniju obezbeđuje kreiranje tabele uz zadavanje broja vrsta i broja redova, a zatim sledi popunjavanje ćelija. Svaka ćelija je, na neki način, dokument za sebe – u okviru nje možete da menjate font, naglašavate slova, centrirate tekst, raspoređujete ga u više redova sa ili bez ravnanja i tome slično. Operacije sa kompletnim redovima su brojne – mogu se menjati dimenzije, kombinovati susedne ćelije u jednu veću ili deliti na više sitnijih, može se uticati na vrstu i izgled linija, pozadinu tabele (tip senčenja) i svašta drugo. Linije se čak mogu sasvim ukinuti, čime se tabela pretvara u nešto sasvim drugo – nekoliko puta su nam zatrebale paralelne kolone (koje inače *WordPerfect* oduvek obezbeđuje) i pokazalo se da je korišćenje tabele sa ukinutim linijama daleko pogodnije rešenje od standardnog mehanizma.

Mora se ipak reći da *WordPerfect for Windows* na polju rada sa tabelama nije doneo mnogo noviteta u odnosu na DOS verziju *WordPerfect* a 5.1 – sve to na ekranu izgleda dosta lepše, a mišem se mogu postići neke nove sitnice, ali je većina opcija identična pa na njih nećemo trošiti previše prostora. Nameće se ipak poređenje sa odgovarajućom opcijom *Venture Professional 3.0* koje ovoga puta ne ide ni u čiju korist: primenjena koncepcija je relativno slična, raspoložive opcije gotovo podudarne, a upotreba jednako jednostavna. Prednost *Venture*-a u tome što se tekst može unapred pripremiti u ASCII formatu (pa, prema tome, i generisati uz pomoć sopstvenog programa) dok *WordPerfect* sav tekst, pa prema tome i tabele, čuva u svom nepojmljivo iskomplikovanom formatu. Zato *WordPerfect* omogućava računanje sa elementima tabele (malo neuobičajeno za tekst procesor, ali zašto da ne?) i direktan uvoz tabela iz programa za unakrsna izračunavanje *PlanPerfect, Lotus, Excel* i *Quattro Pro*.

"Uvoz" podataka se ne svodi na prosto prenošenje – jedan od udarnih noviteta paketa *WordPerfect for Windows* je takozvani DDE (*Dynamic Data Exchange*) Link. Ideja je u tome da podatke pripremate u jednom programu a onda, kada predete u *WordPerfect*, on automatski prenese ispravke na predviđeno mesto. U teoriji izvanredno. U praksi ne baš sjajno, pošto je, pre svega, veza strogo jednosmerna – ako u *WordPerfect* ovoj tabeli nešto izmenite, to se ne prenosi u osnovni program, pa pri sledećem ulasku u *WordPerfect* izmene nestaju. Osim toga, malo je programa koji podržavaju ovakvu razmenu, a i kada pronađete takav, možete da „uvozite" tabele ali ne recimo i „torte" generisane na osnovu tih tabela što je, barem po nama, daleko češće potrebno (zašto bi se izveštaj opterećivao sitnim brojkama kada je grafička prezentacija daleko efektnija?). Uklavno, dobra ideja koje će se, verujemo, razvijati u sledećim verzijama paketa *WordPerfect for Windows*.

NOVI MAKROI NA STARI NAČIN

Čitajući dokumentaciju pomislili smo da je rad sa makroima totalno izmenjen ali se, uz malo eksperimentisanja, pokazalo da se većina stvari radi na stari način. Startujete mod „učenja", zadate ime i (eventualno) kratak opis makroa, a zatim visite neke radnje koje se svode na izbor stavki iz menija, unos teksta i uopšte sve što želite da ubuduće računar sam uradi. Kada završite operaciju, stavite to *WordPerfect*-u do znanja i makro je skoro spreman. *Skoro* spreman, jer će pri prvom izvršavanju biti kompajliran u priličnu neciklu formu koja obezbeđuje znatno brže izvršavanje i ujedno otkriva sve sintaksne greške koje ste napravili. Makro je sada sasvim običan dokument koji možete ispravljati i snimati u samom *WordPerfect*-u, što znači da poseban *Macro Editor* nije potreban – prvi put kada se ispravljen makro bude izvršavao obaviće se njegovo prevođenje pa će sledeći pozivi biti još brži.

Moramo priznati da smo se pitali kako će se u okviru makroa „pamtiti" pokreti mišem i pomalo se iznenadili efikasnim rešenjem koje je firma *WordPerfect* odabrala – nikako. U toku kreiranja makroa miš se, naime, može koristiti za izbor stavki iz menija, ali se pokretanje kurzora mora obavljati sa tastature pa, prema tome, i „pamtiti" na način koji je već uobičajen u prethodnim verzijama *WordPerfect*-a.

Ako se osnovni rad sa makroima nije promenio, složenije naredbe makro jezika su totalno revidirane – sada ih ima daleko više, mnoge službene reči su promenjene, a ima promena i u samoj koncepciji koja je postala pomalo „objektno orijentisana". Na žalost, u okviru dokumentacije makro jezik praktično nije ni pomenut, a čitalac se upućuje na publikaciju *WordPerfect Macro Manual* koja se posebno kupuje, koja nam na žalost (još) nije na raspolaganju. U direktorijumu WPWIN smo pronašli datoteku MACRO.DOC u kojoj su pomalo spartanski opisane makro naredbe tako da smo bili u prilici da napravimo nekoliko umereno složenih makroa – nismo stekli utisak da je čitava operacija išla jednostavnije (ali ni teže) od onoga na šta smo navikli u DOS *WordPerfect*-u 5.1.

Ukoliko ste koristili *WordPerfect 5.1* za DOS i imate dosta makroa za njega, možete ih prilagoditi *Windows* verziji primenom programa MFWIN. Ne može se, na žalost, prevesti sve – dejstvo programa se „završava" kod jednostavnijih makroa koji ne koriste značajne kontrolne strukture, ne rade previše sa *Na... search* opcijom raznih menija i tome slično. Ukratko, pokušajte da konvertujete makroe, ali ne očekujte čuda i pažljivo proverite svaki prevod pre nego što mu poverite neki značajniji tekst.

Nestala je mogućnost da se makroi čija su imena ALTA.WPM, ALTB.WPM itd. izvrše prostim pritiskom na Alt A, Alt B itd. Slična stvar se sada može postići dodeljivanjem makroa tasteru u okviru *Preferences Keyboard* opcije **File** menija. Žalimo za ALTx.WPM makroima jer su bili jako pogodni za prilike u kojima treba na brzinu definisati makro, izvršiti ga par puta a zatim „zaboraviti" – sada svaka takva operacija zahteva nešto više vremena. Za uzvrat, uvedena je mogućnost da makro koji ste definisali praktično pretvorite u naredbu *WordPerfect*-a, ubacite ga u neki meni (standardan ili tasterski) i da ga uopšte daleko kompletnije tretirate.

OSTALI NOVITETI

WordPerfect for Windows donosi i mnoge druge novitete od kojih ćemo pomenuti samo značajnije. Najviše nam se dopala opcija *Auto Code Placement* – njenim uključivanjem „dopuštamo" *WordPerfect*-u da kodove stavi ne tamo gde smo ih mi uneli, nego tamo gde imaju smisla. Sve prethodne verzije su se, na primer, prilično čudno ponašale kada se usred pasusa promene margine, što je kod mnogih početnika izazvalo ne malu zabunu. Ukoliko je *Auto Code Placement* opcija uključena, *WordPerfect* će ovakav kod automatski pomeriti na početak pasusa, čime će anulirati „neuobičajno" ponašanje i ujedno eliminisati zabunu. Ovo je naročito značajno kod rada sa višekolonskim tekstom, gde je položaj kodova izuzetno osetljiv.

Posvetimo malo pažnje i novom editoru za rad sa grafikom – kada u *WordPerfect for Windows* učitate neku sliku (podržani su svi formati koje možete zamisliti, počev od TIFF, IMG i PCX, pa do raznih drugih za koje verovatno niste ni čuli) pruža vam se mogućnost da je pomerate, okrećete, skalirate, ogledalski preokrećete, senčite (njju ili pozadinu) i svašta drugo. I dalje, međutim, nemate mogućnost da crtate ili da delujete na pojedine tačke te slike – za to je neophodno startovati grafički program u kome ste sliku i pripremili. Povoljno je što se slike mogu držati i van .DOC fajla, čime se obezbeđuje da izmena slike bude uvrštena u tekst bez potrebe za njenim ponovnim učitavanjem. Ukoliko slike držite u odvojenim datotekama, nije loše da se te datoteke nalaze u posebnom direktorijumu, a treba paziti i na trenutak kada završeni tekst prenosite na diskete – ne smete da zaboravite prateće slike!

Nastavlja se na strani 70

VENTURA ZA SIROMAŠNE

Microsoft Publisher je deo trojke programskog paketa za _Windows_ okruženje pod imenom _Solutions_, nastao sa zajedničkom idejom vodiljom – da se stvore programi za neke uobičajene oblasti primene, koji ce biti laki za učenje i, kasnije, svakodnevnu upotrebu. _Publisher_ je alat za lako oblikovanje dokumenata, koji od vas ne zahteva da već jeste ili, radeci sa programom, postanete grafičar profesionalac.

Zoran Kohler

Stono izdavaštvo je od dana kada je taj pojam naslao uvek nekako nekako izvan foku sa interesa najvećeg dela korisnika PC racunara. Kada je ta kalegorija primene racunara delimicna korisnici su morali da se suoce sa mnogo ograničenja slabim hardverom sporim računarima i stampacima sa otesko loseg kvaliteta i pratećim slabim softverom – programi koji su imali obraza da se svrstaju u klasu DTP bili su najcešce malo doterani leksi procesori ili prerađeni dinosauri zaostali iz doba obrade teksta na mainframe računarima.

Hardver je napredovao racunari su poslajali sve brzi i mocniji pojavili su se laserski stampači sa cenom koja ih čini dostupnim svima. Softver je takode pratio ovaj razvoj na Irzišu se pojavljuju „Xerox" *Ventura* „Aldus" *Page Maker* i jos nekoliko slicnih pakela za slaganje i pralom teksta. Sve ovo međutim nije ulazak u svet slonog izdavaštva činilo jednostavnijim. Vecina DTP softvera zahteva da svoj ži vot posvetite usvajanju pojmova kao sto su „kerning" i rastavljanje na slogove" ne kaže se bez razloga da je pisanje teksiova u *Word Perfect*-u i slaganje publikacija *Venturs* programom način zivota a ne samo izbor programa. S druge strane klasicni leksi procesori sada vec mogu da u leksi uključuju grafiku mogu se menjati lonlovi slampali u više kolona ali sve to uz mnogo muke oko učenja programa i sticanja iskusiva u radu.

Microsoft Publisher je deo trojke programskih pakela „Solutions" nastalih sa zajednic kom idejom vodiljom – da se stvore programi za neke uobicajene oblasti primene koji ce biti laki za ucenje i kasnije svakodnevnu upotrebu. Publisher je alat za dizajniranja dokumenata koji od vas ne zahteva da već jeslo ili radeci sa programom postanete grafičar profesiona lacD Teslu programa je i pristupljeno imajući lo u vidu priznajem da sa „profesionalnim" programima klase *Ventura* ili *Page Maker* ne mam gotovo nikakvog iskusiva ali imam potre bu za slampanjem dokumenata profesionalnog izgleda. Ono šlo sam do sada uspevao da ura dim sa *Word for Windows*-om (doduse uz jako puno truda) obecava i *Microsoft Publisher* ali uz manje muka. Irebalo je ispitati le tvrdnje.

VERZIJA I INSTALACIJA

Tastirana je verzija 1.00 iz aeptembra 1991 godine. Namojta od ovog programa napisanog za *Windows* okruženje očekivali da vas impresionira pakelom u kome se nalazi desetak diskela i dvadeselak centimelara debelo upulstvo program dolazi na jednoj diskeli formala 5.25"/1.2MB u pakelu sa dobija i di skela formala 3.5" što je uobičajeno za većinu novih programa.

"Microsoft" priručnici uvek su vrlo iscrpni dobro dizajnirski rešeni i laki za čitanje i snalaženje. Ni „Publisher Users's Guide" nije izuzelak – priručnik je doduše lanji od uputstva za *Windows* ali zbog loga nije manje korisian. Vrlo je detaljan i podeljen na poglavlja za brzo ucenje programa i ona za naprednije korsnike. I ovde do izražaja dolaze dobre osobine *Win dows* pakela, koje omogućavaju da se i korsnik i programer aplikacije posvele jednom konkretnom problemu a u okruženje se mogu pouzdali u vezi sa svim ostalim načinom akli viranja programa, načinom izdavanja komandi procedurama za kretanje po dokumenlu i slič no.

Publisher zahteva IBM AT ili PS/2 klase tvrdi disk sa najmanje 4MB slobodnog prostora (za punu instalaciju programa). 1MB ili više RAM memorije. *Windows* 3.0 ili noviji MS DOS ili PC DOS 3.1 ili noviji, monitor i grafičku kartu podržanu od strana *Windows*a 3.0. Mis se preporučuje, radi lakoće kor šcenja programa, a praktično je neophodan.

Za punu instalaciju programa potrabno je oko 4MB slobodnog prostora na disku. Iz **File**–**Run** menija *Windows*-a pokrenucela pro gram SETUP sa instalaciona diskele. SETUP će proveriti da li na disku ima dovoljno slobod nog prostora. Moguće je instalirati i samo deo paketa, ili naknadno instalirati delove koja ste prvi put ispustili. Moguce su dve su opcije **Complete Installation** i **Custom Installation**. **Complete Installation** će ceo pakel instalirati na C disk bez mnogo dijaloga sa konsnikom. **Custom** opcija omogucava vam da *Publisher* instalirate na neki drugi disk, ili da na disk pre nesete samo dalova pakela.

UVOD U *PUBLISHER*

Kada se zavrsi instalacija *Microsoft Publis her* nudi da se uđe u zanimljiv „Introduction to Publisher" (Uvod u *Publisher*) deo koji korisni ka brzo provodi kroz razne faze dizajna publi kacije, pokazuje mu sposobnosti programa i upoznaje ga sa osnovnim konceptima rada.

Publikacije se tradiocionalno kreiraju slaga njem objekata na listu papira – stari dobri cul and paste. Pored svog naprelka tehnologija i deo izrade casopisa koji upravo čitate upravo je takav (naravno na voljom uredništva). Rad u *Publisher* u jako je sličan kao i ranije, kreiranje i kombinujete razne objekte, ali sada u njihovom elektronskom obliku. *Publisher* poznaje sedam vrsta objekata i za svaki vezuja odgovarajući alat. Alati su ikona BUTTON tipa ispod osnovne linije menija.

Publisher je smišljen na konceplu „okvira" pravougaonih površina koje sadrža teksl ili ilu stracije. Za razliku od teksta u leksi procesori ma ovde tekst može da postoji samo unutar okvira – FRAME. Okvin se tekstom kreiraju se TEXT FRAME alatom. Ilustracije posloje u okvirima za slike. Vezanost za okvire omogu ćava korisniku da lako menja izgled dokumen ta. Ukratko da biste kreirali bilo kakav objekat, kliknucete na ikoni alata na listu misem označi meslo objekta ! – gotovo. Objektu zatim treba dodeliti teksl, grafiku ili neki drugi sadrzaj.

Rad na novoj publikaciji u *Publisher*-u može početi na tri načina: otvoriti neki ranije završen dokument i raditi na osnovu njegovog sadrzaja, početi od prazna strane koristiti predloške (TEMPLATES) za veliki broj razlici tih dokumenata, ili prapustiti posao oko defin isanja osnovnog izgleda vasag remek dala *Pa ge Wizard*-u koji ce sam uradili veliki deo po sla umeslo korisnika.

DA LI JE ZAISTA LAKO?

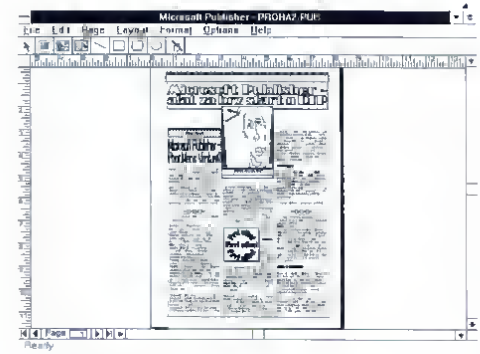
"Microsoft" se zaista potrudio da *Publisher* bude lak za upotrebu. Dovoljno je imati izve sno iskusivo u radu sa mocnijim leksi procesor om i *Windows*-om da bi se mogao započeti rad sa programom. Naravno to je jos jedan dokaz da je ideja *Windows* okruzenja bila pravi pogodak – *Publisher* poznaje iste sekvence pokrela i komandi misa iste skracene komanda da sa laslalure i pokazuje standardne osobina *Windows* aplikacije (svakako i lose strane mnogo prostora na disku, sporost u nekim operacijama...).

Da bisle složili leksl koji zadovoljava neke osnovna kriterijume grafičkog dizajna morale ipak poznavali osnove zanala. Uputstvo za *Pu blisher* ovde je od pomoći jer na jednostavan nacin objašnjava stvari od značaja: izgled stra ne, izbor objekata na strani, nacin isticanja vaznih delalja i slično. Od pomoći su i uzorci raznih vrsla dokumenala, koje vrlo brzo nauci ta da menjate prema vašim potrebama. U veci ni dokumenata koje *Publisher* predlaze dovolj no je promeniti naslove i slike uz leksl zamе nili leksl i vrlo brzo imali sve spremno za šlam panje.

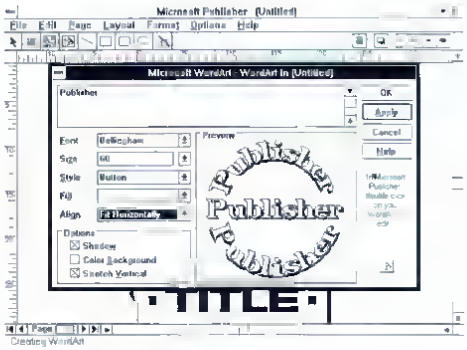
NAJBOLJE JE PROBATI

Nisam stri cnjak za stono izdavaštvo i do sada mi je najveći domaći na tom polju bio of krivanje naprednih mogucnosti programa *Word for Windows*. Zalo je verujem najbolji nacin da se ispita *Microsoft Publisher* bio da se pro ba neslo ko nkretno. Kao primer izabrao sam jednostavan leksl – prvu varziju ovog clanka sa skromnim zahtevima u pogledu dizajna. Na sličan korak ce se sigurno odlučili gotovo svaki novi korisnik ovog programa.

Veliku pomoć u ovakvom poslu čini PAGE WIZARD – skup rutina ili bolje receno makroa za kreiranje nekoliko standardnih publikacija. Čarobnjak zna sam da stvori sedam osnovnih kalegorija publikacija formulare za poslovna primene (FAX obrazac, narudzbenica, račun razni spiskovi), bilten (NEWSLETTER), brosu ru i propagandni malenjal, čestitke i pozivnice za zabavu i veselje primene. *Word* za kreira kalendare. I avione od papira. Kod avione od papira kao što možete prelpostaviti PAGE WIZARD se ograničava na štampanje nacrta po kome cete savijali papir i na isp sivanje oznaka na krilima: suvise je očekivali da je *Mi crosoft* smišlio nacin da laserski stampač pravi avione – ipak je to oblast u kojoj je vrlo uspe šan Popravnik Peca junak iz strip a.



Izgled prelomljene strane na ekranu



***WordArt*- Alatka za manipulisanje tekstom**

Za nasu probu slaganja jednog teksta profesionalnog izgleda izabrane je opcija NEWSLETTER odnosno bilten iz PAGE WIZARD-a. Izborom ovog tipa dokumenta zadajete *Publisher*-u zadatak da kreira dokument koji liči na novine glasnik kompanije ili mali magazin. Ovde kao i za sve ostale vrste dokumenata, možete birati tri stila klasičan moderan i „jazzy" – pomalo avangarden. Na ova pitanja prvi put možete odgovoriti i na slepo jer se možete pouzdati da svaka varijanta izgleda dovoljno dobro. To nikako ne znači da je program *foolproof*, odnosno otporan na greške korisnika; *fool-proof* programi obično završe svoj razvojni put kao programi prilagođeni lošijim laicima i time obeshrabre korisnike od eksperimentisanja i učenja naprednih tehnika. *Publisher* ni po čemu ne spade u tu klasu, jer je pomoć koju vam pruža velika, ali takva da oslavlja dovoljno mesta za vaše lične eksperimenta.

U prvih par koraka PAGE WIZARD postavlja par pitanja a zatim pretazi na delo pred vašim očima odvija se mala animacija, dok program izvodi seriju operacija neophodnih za nastajanje dokumente. To je prvo definisanje dimenzija strane, postavljanje vodica za slaganje objekata, kreiranje pozadine svake strene – sadržaja koji se ponavlja na svim stranama, definisanje objekata za tekst i ilustracije i na kraju, vezivanje primera teksta i slika za objekte (okvira). Za sve to vreme aktivan je dijalog prozor, u kome možete menjati brzinu formiranja publikacije. Ako program usporite jako cete pratiti sled koraka koji dovode do kreiranja kompletne publikacije i iz toga dosta nauciti. Medutim, kako je ovo prvi ozbiljniji susret sa programom, pusticemo da se procedura odvija punom brzinom i samo sačekati razultate.

Rezultat rada čarobnjaka je glasnik sa jekstom koji je ju samo za primer, a na vama je da ga zamenite pravim. Poslupak je jasno objasnjen u uputstvu, ali je i intuitivno lako zaključiti šta treba uraditi. Izabrati jedan od teksl okvire. Aktivirati komandu **File...Import Text**. Tekst se može nalaziti u drugom *Publisher* dokumentu (sa ekstenzijom .PUB), običnoj jekstualnoj datoteci, ili u datoteci koja je rezultat rada nekog od direktno podržanih tekst procesora: *MS Word*, *MS Works* i *MS Works for Windows*, *Word Perfect* 5.0 i 5.1, *Word for Windows*, *Wordstar* 3.3 do 5.0 i u *Rich Text Format*-u (RTF). Program automatski prenosi dugacak tekst u sledeci slobodan tekstualni okvir, ili to činile sami ako želite da postignete specijalna alakte (prenos teksta izmedu nesusednih strana). Takst u okviru formatirate na način koji vam je poznat iz drugih programa na raspolaganju su razne opcije poravnavanja teksta bold otisak, italika i ostale varijante slo ge. Ako pišete na engleskom korekhnosi reči možete proveriti SPELLING CHECKER-om a oko razdvajanja raci na slogove pomoci će vam naredba za raslavljanje na slogove (HYPHENATION).

U *Microsoft Publisher* u naći ćele i dodatke kojih nema u drugim DTP paketima ili zahteveju dugacak i kompikovan rad. Jedna od posebnih mogućnosti jasu WORD ART objekti tekst formatiran prema knvama iskrivljen ili ispisan naopako sa elektima senke, raznih boja i rastezenjem po horizontali i vertikali. U WORD ART-u na raspolaganju nam stoji veliki izbor fontova ali naravno bez naših znakova. To svakako smanjuje korisnost ovog dela *Publisher*-a za kratko vreme za koje mi je na raspolaganju bio program, nisam uspeo da olrjem način na koji bi se dodali fontovi sa znacime jugoslovenske latinice i ćirilice. Verovatno da ta mogucnost postoji ali za to treba više vremena provesli u radu sa programom.

Toliko o osobinama *Publisher*-a, vratimo se probnoj publikaciji. PAGE WIZARD kreirao je dakle, okvire za **tekst** i slike. Da bismo ubacili naš tekst, moramo prvo ukloniti takst koji je čarobnjak uneo kao primer. Okvir se selektuje a zatim izda naredba **Edit...Delete Text**. Nakon toga u okviru za dijalog naredbe **File...Import Text** valja izabrati daloteku obicnog ASCII formata i kliknuti na imenu u listi daloteka. *Publisher* tekst iz daloteke smešta u izabrani okvir ako je tekst duzi nudi mogucnost automatskog prenosa u sledeci (susedni) okvir. U vama ostavlja da sami oznacite u koji ce se okvir leksl prenet. Ova druga varijanta je bez bednija ako strana sadrži dosta okvira koji va ma samo na izgled ne izgledaju kao susedni.

Zamenjen je tekst, ostalo je još da se promene naslovi, dodaju potpisi ispod slika koje je u dokumenti postavio PAGE WIZARD, malo promeni raspored stubaca – i prva verzija publikacije bila je gotova. Vežbe radi oddao sam brojeve strana, kao probu mogućnosti rada sa objektima koji se ponavljaju na svim stranama, a zatim menjao raspored objekata. Sve to da sa uradi bez mnogo napora, samo sa znanjem slečenim u radu sa teksl procesorima i u konscenju *Windows*-a. Ceo posao, koji se može shvatiti i kao učenje programa, oduzeo je samo nekoliko sati. Ostalo je još samo da se ispitaju mogućnosti kontrole štampanja, koje su zaista velike. Tast publikacija nije zahtevala nake izuzetne opcije za štampanje, najviše zbog izabranog tipa (NEWSLETTER). *Publisher* nudi razne predefinisane i posebne tipove štampanja: stampanje dve strane na isloj strani, lisla, štampanje čestitki, obična štampa, štampanje knjiga uz vodenje računa o prostoru koji se ostavlja za savijanje i končenje, i slicno.

ZAŠTO JE DOBAR

U konščanju *Microsoft Publisher*-a pomalo smetaju nekoliko stvari – jedna od njih je nedostatak mogucnosti da se u delu WORD ART definišu naša slova, a druga je sporost u radu u nekim situacijama. To dolazi do izražaja kada se na strani nalazi veliki broj objekata. I kada veliki deo tih objekata čine slike. Tada dolazi do izražaja bromost *Windows*-a koja se javlja kada su resursi računara – memorija i disk – maksimalno opterećeni.

Microsofl Pubisher je zanimljiv, a pra sve ga dobar i koristan program. Interesantan je i zalo što uključuje OLE (Object Linking and Embedding) tehniku najavljenu za novi *Windows* 3.1. Kada je odbijen na testiranja zamišljen je zaista tezak zadatak: da ispuni potrebe za kreiranjem dokumenata profesionalnog izgleda rukama korisnika koji se nije ranije iskušao na programima kao što su „Xerox" *Ventura Publisher* ili „Aldus" *Page Maker*. Taj je zahlev ispunjen na zaista zadovoljavajuci način. *Publisher* od korsnika zaista zahteva više truda nego neki od teksl procesora, ima i nešlo manje naprednih mogucnosti nego *Ventura Publisher*. No, ako je kriterijum da treba bez nekoliko nedelja učenja rada na komplikovanom DTP paketu i teorije grafickog dizajna sastaviti dokument prsloinog izgleda, ovaj paket svakako zaslužuje vašu pažnju. Izdatak za njega brzo će se vratiti a „Microsoft" je kvalitetom i povoljnom cenom (oko 150 USD) pobližo stono izdavaštvo i korisnicima kojima je ono samo povremeno potrebno.

Zahvaljujemo se na kraju kompaniji „Bel Pagatte" iz Beograda koje nam je paket *Microsoft Publisher* ustupila na testiranje.

Kontakt adresa

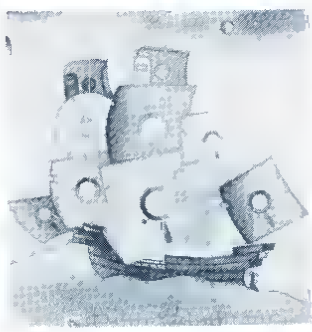
Microsoft GmbH
Edisonstrasse 1
D 8044 Unterschleissheim, Deutchland
Tel 9949 89 3170 5-0
Fax 9949 89 3170 5-100

UBRZANE TABELE

U prolece prošle godine imali amo zadovoljsvo da se upoznamo sa *Baler*-om, prvim kompajlerom za *Lotus 1-2-3* i kompatibilne tabelarne programe. Pošto tržište ne miruje, u meduvremenu je objavljeno još nekoliko novih proizvoda iz ove oblasti. Kao veoma ambiciozni proizvodač, pojavila se firma „Sheng Labs" iz SAD, sa dva paketa: *King Jaguar*, kompajler za radne tabele (.WK1) kompatibilne sa *Lotusom 1-2-3* do verzije *2.2* i *Composa*, prvi kompajler za *Lotus Symphony*.

Vladimir Stamenović

Obzirom da oba paketa čekaju na testiranje još od polovine prošle godine, pretpostavljam da je *King Jaguar* (*KJ*) do sada azuriran u odnosu na novi *Lotus 1-2-3* ver. *2.3*. Povoljna okolnost je što se mogu kompajlirati sve datoteke tipa .WK1, a znamo da ih možemo dobiti i u *Quattro Pro*-u i u *Excel*-u i masi drugih programa za tabelarna izračunavanja. Naravno u tim slucajevima treba voditi racuna da se u tabelama koriste samo one funkcije i makro komande sa kojima raspolaze *Lotus 1-2-3*.



LOTUS BEZ LOTUSA

Šta je to što nuda kompajleri za radne tabele? Osnovna stvar je da možete napraviti aplikaciju u radnoj tabeli a zatim, uz pomoc kompajlera, i njenu EXE verziju. Iako da se može pokrenuti bez *Lotus*-a *1-2-3* ili kompatibilnog programa, znaci direktno iz DOS-a. Postupak je takav da se u toku kompilacije prvo vrši analiza sadržaja kompletne tabele sa svim makroima, formulama i podacima. Ako je sve u redu vrši se pravodanje u mašinski kod. Pri čemu nastaje jedna JWK datoteka za svaku WK1 datoteku. Za startovanje aplikacije osim JWK datoteke neophodan je i pokretacki program *runtime engine*. To je izvršni program (.EXE) koji sadrzi sve što je neophodno da može da pokrene našu aplikaciju. Mada se kod nas niko ne pridržava propisa o zaštiti autorskih programerskih prava, znacajno je i to da se pokretacki program može distribuirati dalje korisnicima bez ikakvih obaveza prama proizvodaču.

Dakle upotrebom kompajlera dolazimo do nekoliko lepih pogodnosti i mogucnosti. Prva aplikacija koju smo napravili i u koju smo uložili veliki rad i trud je, nakon prevodenja apsolutno bezbedna od neovlascenog kopiranja i korišcenja naših makro kodova, formula i slicno, tako da smo u situaciji da prodamo takvu aplikaciju bez brige o zloupotrebi. Drugo moguce su velike ušteda na kupovini softvera. Naime za rad sa kompajliranom tabelom nam nije potreban originalni *Lotus 1-2-3* ili slican program. To znaci da u jednoj firmi možete imati samo jedan *Lotus 1-2-3* i samo jedan *King Jaguar* pomoću kojih cete napraviti neograniceni broj aplikacija i dalti ih korisnicima. Trece *King Jaguar* nudi dodatne mogucnosti za dotevnanje aplikacije koje inace, nisu raspoložive u *1-2-3*. Ovaj deo je prilicno zanimljiv jer je dodato dosta novih makro komandi kojima je moguce kontrolisati izgleda bas sve elemente na ekranu, ukljucujuci boje, indikatore korisnicke poruke, poruke o greškama itd. Osim toga, pomocu novih makro komandi mozete napraviti sopstvene menije (vertikalne i pop-up), kontekstno osetljivi help, kontrolisati unos podataka i još mnogo toga.

NA C-U I ASEMBLERU

Kao i *Baler*, i *King Jaguar* podržava UDF ove (*User Defined Functions* – funkcije koje kreira sam korisnik). Pomocu njih korisnik može praviti aplikacije skoro bez ikakvih ogranicenja, koja inace posloje u *1-2-3*. Za pisanje UDF ova možala koristiti *Mcrosoft C* verzija *5.1*, *Quick C* verzija *2.0* ili asembler. No ako nisle programer, na raspolaganju je gotova bibioteka korisnickih funkcija koje ce korisno poslužiti u dotervanju vase aplikacije. Njihova upotreba se na razlikuje od upotrebe klasicnih makro komandi. U upuistvu za upotrebu *King Jaguar* a je dat spisak sa opisom raspoloživih funkcija. Možete, cak napraviti takve UDF-ove koji ce ompouciti vezu vaše aplikacije sa razlicitim mamim i drugim instrumentima, sa eksternim bazama podataka, kao i kreiranje dodatnih (add in) programa razlicitih namena.

Vrlo korisna i zanimljiva je i mogucnosi da se iz pokretackog programa izbace opcije koje *1-2-3* normalno sadrži a koje nam nisu potrebne u aplikaciji. Time se postiže manja velicina pokretalčkog programa, onemogucuje se korisnicima da rade ono sta nije predviđeno aplikacijom, dok istovremeno makro komande imaju pristup i do ith isključenih" c komandi. Znaci, ako na primer, našom aplikacijom nije predviđeno da se prave grafikoni iz pokretalckog programa cemo isključiti komplalnu */Graph* opciju sa svim pripadajucim komandama. Identicna mogucnosi postoji i u *Baleru*, i ja je ocanjujem vrlo korisnom.

King Jaguar ne podrzava *Allways*, dodatak za stampu za *1-2-3* verzija *2.2* kao ni druge dodatne programa koji se mogu koristiti uz *1-2-3*. Možda ce vam to izgledati malo nezgodno pogotovu ako ste navikli na upotrebu nekog od njih. Medutim olakšavajuca okolnosi je što *King Jaguar* sadrži nekoliko interesantnih dodatnih programa. Pre svih tu je *Macro View* makro dibager. Slicno dodatnim programima za *1-2-3* ovog tipa o kojima sam pisao. *Macro View* omogucuje lemejno ispitivanje funkcionalnosti i ispravnosti makro procedura iz vase aplikacije. Tu možete odrediti i tacka prekida izvršavanja makroa (*breakpoints*) i pratiti ponašanje promenjivih velicina kroz tacke pregleda (*watchpoints*). Osim toga *Macro View* slicno *Audilor*-u iz *1-2-3* ver. *2.3* omogucuje pracenje izracunavanja pojedinih a medusobno za visnih formula. Korisno!

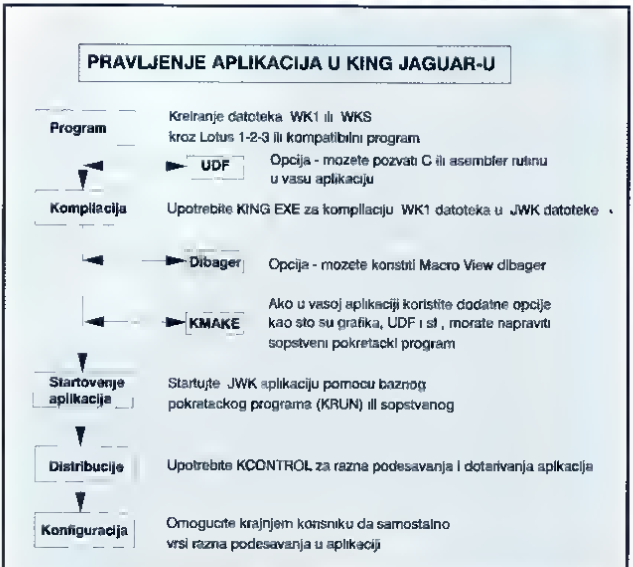
Kada napravite aplikaciju u radnoj tabeli vrlo lako je mpouce da se u makro kodu provuce naka greška jer *1-2-3* ne prijavljuje grešku u sintaksi sve dok se svaki deo procedure ne izvrši. Oluda se dešava da narzgled korektno napisana procedura kasnije prouzrokuje prekid programa. Prilkom kompilacije, *King Jaguar* proverava svaku celiju u svim datotekama iz aplikacije pa ja u slanju da otkrije sve sintaksne greške. Znacajno je i to da *King Jaguar* odzvoljava linkovanje podataka iz više tabela, kao i kombinovanje delova tabela u jednoj (*File Combine*). Cak šta više, prilkom kompilaci je cim naide na makro ili formulu koje se pozivaju na podatke iz druge tabele, izvrši ce i njihovu kompilaciju. Ukoliko se pri tome pojavi greška li više njih, mozate ih zapisati u datoteku i na lanane prostudirati.

Interesantno je da *King Jaguar* sadrži i mogucnosi stampanja „caršava" odnosno velikih tabela uzduž papira za printer. Konkretno ta opcija je indenticna onoj koju nudi *Sideways*, dodatni prporam za *1-2-3*, o kome smo vec pisali. Isto tako pri pravljenju sopstvenog pokretalckog programa možete zahtevati upotrebu koprocesora (8087, 80287 i 80387) s tim da treba imati na umu da je onda stvarno potreban koprocesor za rad programa.

POKRETAČKI PROGRAM

King Jaguar radi na svakom IBM PC racunaru sa najmanja 512 K RAM-a. Probao sam ga i ugla vnom, ali cno mada sam nailazio na probleme sa manjkom RAM-a pri kreiranju sopstvenih pokretalčkih prporama. Morate imati i instaliran DOS 2.0 (3.1 za rad u mreži) ili noviji. Kao sto sam vec rekao, možete vrsiti dotervanje i kompilaciju tabela koje su napravljene u bilo kom programu koji može da proizvede .WK1 datoteke, a kompatibilan je sa *Lotus*-om *1-2-3* verzija *2.2*. Što se iice memonje *King Jaguar* automatski koristi *expanded* memoriju do 8Mb sa LIM 3.2 drajverom, odnosno do 32 Mb sa LIM 4.0. Ako nemate *expanded* memoriju *King Jaguar* ce koristiti prostor na cvrstom disku kao *expanded* memoriju. Naravno, mo guce je odrediti deo diska, odnosno direktorijum koji ce biti u upotrebi za tu svrhu.

Da vidimo kakav je *King Jaguar* u akciji. U svrhu testa napravio sam nekoliko standardnih datoteka sa test tabelama. Na primer, u jednoj se nalaze sve raspoložive funkcije iz *1-2-3* i služi mi za ispitivanje kompatibilnosti i poredenje sa drugim tabelarnim programima. U drugoj pak imam jednostavnu aplikaciju sa menijem, a u upotrebi je dosta razlicitih makro komandi, kroz nekoliko procedura. Namena takve aplikacije, u ovom slucaju je da vidim kakva su mogucnosti dotervanja kroz *King Ja*



guer kekva je brzina kompajliranje Je kojom brzinom se kompajlirana tabela izvrsava u odnosu na normalnu iz *1-2-3* Sve probe i testove sam radio na IBM PC/AT 286 recunaru sa takt lom 12 MHz

Prvo sto treba uradili je da proverite da li treba da napravite sopstveni pokretacki program Naime uz *King Jaguer* dobijate gotov program KRUN EXE kojim možete startovati kompajlirane aplikacije ali on ne podržava sve raspoložive opcije Konkretno ako vam u aplikaciji treba grafika korišcenje koprocesora rad sa malncama ild onda KRUN nece zadovoljiti vaše potrebe Dakle pomoću programa KMAKE treba generisati sopstveni startni program koji cete nazvati onako kako vam odgovara i u kome cete naznaciti šta on ave treba da omogući Obzirom da se u mojoj testaplikaciji podeci mogu prikazati gralicki neophodno je bilo da napravim sopstveni pokretacki program a nazvao sam ga KJPROBA EXE Iznenadila me je brzine kojom je program kreiran jer u *Baleru* to dosta duze traje KMAKE je generisao KJPROBA EXE za samo 2 5 minuta! Pn tome je napravio i datoteku KJPROBA SET iz koje pokretacki program cita neka osnovna podešavanja kao što su direktorijum za virtuelnu memoriju boje elemenata na ekranu štampanje uzduz papira ild Za promenu tih osnovnih elemenata konstlicemo program KCONTROL On kroz interaktivni rad omogucuja jednostavno dotenvanje aplikacije Lepo je rešen izbor boja za sve elemente ekrana a vrlo lako možete iskljuciti nepotrebne opcije iz klasicnog *1-2 3* menija Time korisnike onemogucujete da preduzimaju nepredvidene akcije ali makroi i dalje imaju pristup svim opcijama

Sve izmene koje napravimo u SET datoteci ce vaziti i za druge aplikacije koje pokrecemo istim pokretackim programom inače *King Jaguar* je predvideo i mogucnost da krajnji korisnik dekle onaj kome smo mi isporucili kompajliranu aplikaciju može da menja osnovno podešavanje i to kroz program KCONFIG

Moj program za startovanje aplikacija je bio dosta veci od originalnog (KRUN EXE = 361903 a KJPROBA EXE – 477241 bajtova) što je i razumljivo jer sadrži znatno vise mogucnosti U vezi s tim moza vam se desiti da radite aplikaciju za korisnika koji ima PC/XT sa flopjima od 360 K U tom slucaju, pn kreiranju pokretačkog programa zadaćete opciju SPLIT koja ce podeliti ovako veliki program u dve delа tako da slanu na diskete od 360 K U mom primeru KMAKE je napravio dve daloteke KJPROBA EXE (237056 bajtova) i KJPROBA OVL (240185 bajtova) a posao je trajeo oko 4 minuta

JAGUAR U AKCIJI

Dakle kada smo napravili sopstveni pokretački program i udesili ga po nasoj želji treba kompajlirati originalnu radnu tabelu Poslupak je jednostavan Pokrenućete program KING kome cete dodati ime WK1 daloteke koju kompajlirate U mom slucaju

KING TESTMKR1

Ako startujete samo KING program onda cete moći kroz meni da odaberete datoteku za prevodenje KING kreim JWK daloteke koje odgovaraju originalnim WK1 datotekama

Kompilacija je izuzelno brza opet u poredenju sa Balerom Naime daloteka sa makroima (bez ikakvih podataka oko 5 K) je prevedena takoreci u sekundi dok je kompajliranje datoteke sa funkcijama (oko 100 K) trajalo samo 1 5 minut. Zaista brzo

Pn kompilaciji se desevaju i neke zanimljive stvari Na primer. *King Jaguar* uklanja sve prazne celije iz aplikacije da bi smanjio velicinu deloteke Ako imate potrebu da unosite podatke u takve celije nacela to moci eko pre kompilacije ne skinete zaštitu sa tih celija On da ih *King Jaguar* nece diratl Dalje oblasti (*ranga*) celija koje sadrže makroe formule i funkcije su nakon kompilacije nedodirljive odnosno u tim celijama se ne može ništa raditi (**/Copy, /Move, /Range Erase**) I ne samo to – sadrzaj celija sa makroima formulama je nevidljiv i tako apsolutno zaštican Ako se u aplikaciji koriste podaci iz više daloteka *King Jaguar* će ih sve koji se spominju automatski kompajlirati I na kraju kao što sam već napomenuo pnlikom prevođenja *King Jaguar* ispituje sve celije bez obzira šta one sadrže To je dosla zgodno jer tom prilikom se mogu otkriti na primer greške u sintaksi u makroima koje niste primetili u originalnoj tabeli pre kompilacije

Na kraju ostalo je de pokrenemo prevedenu aplikaciju To se radi tako sto se startuje pokretacki program (originalni ili sopstveni) kome se doda ime kompajlirane tabele (JWK daloteka) U mom primeru to bi bilo

KJPROBA TESTMKR1

Startovanjem samo pokretačkog programe KJPROBA mogao bih da kroz meni odaberem neku (ako ih ima više) JWK daloteku sa kojom hocu da radim Moguće je i uprostiti stvar tako što cemo i pokretackom programu aplikacije dati isto ime Onda je dovoljno da pozovemo program po imenu i on ce se kompletno izvršiti Takvu situaciju treba u napred predvideti pri kreiranju sopstvenog pokretačkog programa jer prostim primenovanjem daloteka nacemo uraditi posao

Krajnjim korisnicima se isporučuju celiri da loteka

- kompajlirana tabela (JWK daloteka)
- pokretacki program (na primer KJPROBA EXE ili originalni KRUN EXE)
- sal daloteka za pokretacki program (na primer KJPROBA SET)
- konfiguraciona datoteka (KCONFIG EXE)

DOBRE APLIKACIJE

Na kraju evo nekih utisaka Kod *King Jaguer* ima i dobrih i loših stvari U dobre bih ubrojao to što po mom ušsku pravi verovatno najbrže samostalne aplikacije baznrane na makroima iz *1-2 3* Čak brže nego *Baler* U mojim testovima samostalne aplikacija je radila 4 do 6 puta brze nego u okviru *1-2-3* verzija *2 2* Svida mi se i to sto ima sasvim dobar makro dibager a pogolovu to što mozete dotenvati aplikaciju po želji skoro bez ograničenja pravec procedure u C-u ili asambleru U loše uslovno receno stvari bih ubrojao to što veliki broj konsnih makro komandi koje su u saglasnu *King Jaguar*a morate unositi ručno" u aplikaciju za razliku od *Bafera* gde je to išlo interaktivno. kroz menije Dodao b h i izvesnu ne kompatibilnost sa *1-2 3* mada su svi takvi slučajevi detaljno nabrojani i opisani (izuzev **TABLE**) komanda) dekle lako se mogu izbeci

King Jaguar je dobar program namenjen onima koji prave aplikacije za druge korisnike a da li je bolji od drugih te vrste (*Baler*) nisam u stanju da odgovorim U svakom slučaju onaj ko bude s njim radio a podrazumeva se da dobro zna programiranje u *Lotus-u 1-2-3* moći će da napravi zaista dobre aplikacije

Korisne adresa

Shang Labs, Inc
4470 SW Hell St #282
Beaverton, OR 97005-9919 U S A
Tel (503)646-3894

ZA LJUBITELJE MATRICA

U inženjerskim i naučnim primenama najčešća struktura podataka su matrice. *Gauss* je prototip matričnog jezika – jezika koji osim brojevima i tekstom operiše još samo sa matricama. Ima iste osnovne operacije kao i bejzik, uz nešto izmenjenu, ceoliku sintaksu. Međutim, za razliku od bejzika, *Gauss* podržava stotine naredbi za rad sa matricama.

Đorđe Savić

Programski jezici bazirаju se na tipovima podataka i operacijama koje se nad njima mogu izvršiti. Uobičajeni tipovi podataka su brojevi, slova, tekst (*string*), nizovi, matrice, slekovi, redovi čekanja. Jezici opšte namene (bejzik, C, paskal) delimično samo osnovu jezika s tim da neko drugi isporuči module, potprograme, funkcije, objekte za preko potrebne programerske operacije (prozori, komunikacije, rad sa diskom, baze podataka i druge). Problemski orijentisani jezici usredsređuju se na posebne oblasti i oslanjanju se na strukture podataka koje su za zadatu oblast karakteristične.

PAKET I INSTALACIJA

Gauss se isporučuje na jedanaest disketa promera 5,25 inča i kapaciteta 360K. Za rad su mu potrebni DOS 2.0 ili noviji, tvrdi disk i matematički koprocesor. Umesto hardverskog koprocesora možemo upotrebiti softversku emulaciju koprocesora u vidu *public domain* programa po imenu EM87. *Gauss* smo testirali na prosečnom AT računaru današnjice, to jest na 12MHz bez stanja čekanja. Brzina je bila zadovoljavajuća, a sigurno bi odziv sistema bio trenutan da je u sistemu postojao hardverski koprocesor.

Na disketama je generička verzija *Gauss*a, to jest verzija koja radi na svim PC računarima od procesora 8088 pa naviše. Od proizvođača je moguće dobiti posebne verzije za 80286/80287 računare, kao i za kombinaciju 80386/80387. Na 386 mašinama sve se dogada čak 30*% brže jer *Gauss* tu radi potpuno 32-bitno.

Priručnik je niz od 730 strana uvezanih trostrukim metalnim prstenom u tvrde plastične korice. Podeljen je na logičke celine, ali je uvod u rad sa programom (*tutorial*) poslavljen na sam kraj, a instalacija nije obrađena na jednom mestu, nego za svaku celinu posebno. Ako ne želimo nepotrebne delove programa ne moramo uopšte ni instalirati. Nedostatak ovog pristupa instalaciji programa je u tome što kod čitanja svakog novog poglavlja moramo umetati diskete, menjati AUTOEXEC.BAT, deloteku i resalovati sistem.

Instalaciju najlakše obavljamo isporučenim programom GINSTALL, a možemo i „ručno" naredbom COPY iz DOS-a. Posle instalacija na disku ostaje imenik C:\GAUSS sa podimenicima za tekst programa (procedura (SRC), bibl-oteke (LIB), primere (EXAMPLES) i razne izvršne programe. U AUTOEXEC.BAT dajdeloteku moramo dodati niz SET naredbi sa promenljivama koja usmeravaju na pojedine od ih podimenika. *Gauss* zauzima preko tri megabajta.

Instalacija bi mogla biti i bolja a priručnik bi mogao biti preglednji. Nedostaje tekst koji bi korisnika motivisao na kreativnu upotrebu paketa. Listinga procedura na *Gaussu* ima na stotine kilobajta, ali nije dat nijedan kompletan primer „iz života" recimo kako rešavati jednačine matematičke fizike, kako upotrebiti *Gauss* u statistici i slično. Verovatno se pošlo od pretpostavke da ko kupi *Gauss* ionako zna koje će probleme rešavati i kako će ih matematički prikazivati.

RAD SA *GAUSS*-OM

Gauss ima dva editora. Po učitavanju *Gaussa* u memoriju korisnik je u linijskom editoru, a naredbom EDIT odatle se može prebaciti u potpun ekranski editor. Linijski editor je takođe i ekranski jer možemo pomerati kursor po ekranu i aktivirati naredbe koje vidimo na ekranu. Naredbe ekranskog editora nimalo ne liče na naredbe *WordStara*, što moderni korisnik shvata kao napor a nikako ne kao vrlinu. Editori u *Gaussu* ponajviše podsećaju ne prve bejzike na PC računarima, ali danas – osim što su upotrebljivi u osnovne svrhe – nikakvih drugih kvaliteta nemaju.

Gauss je interpreter visokog nivoa, jer zadate naredbe prvo prevodi u jednoznačne simbole da bi ih kasnije, u drugom prolazu, izvršavao direktnim pozivima potprograma. Ti potprogrami su ručno optimizovani u asembleru i odgovaraju ugrađenim naredbama jezika za matrične i tekstualne operacije. Programi u *Gaussu* su zato mnogo brži od ekvivalentnih programa u programskim jezicima opšte namene. Dalo je i nekih 120 potprograma u izvornom obliku, što jasi u vidu teksle programa na *Gaussu*.

NALIK NA C

Gauss je interaktivan po ugledu na bejzik ali mu sintaksa liči ne C. Na primer, komentar je svaki tekst između simbola /* i */, potprogrami se zovu funkcije jer svaki vraca razultat i slično. Program na *Gaussu* je niz naredbi međusobno razdvojenih tačkom i zarezom. Znak pndodeljivanja je = a ne := kao u paskalu. Sledeći red sadrži tri naredbe pndodeljivanja:

```
x = 5  z = rndn(3,3)  y = x + z
```

Promenljiva se deklariše u skladu sa vrednoscu na desnoj strani znaka =. Promenljiva **x** ima vrednosi pel, **z** dobija vrednosi slučajne promenljive ganarisane po normalnoj raspodeli (to je uloga funkcije **rndn**). Promenljiva **y** je zbir od **x** i **z**. Naredbom

```
prini x y z
```

pnkazujemo ove vrednosti na ekranu.

Naredba može imati obeležje i posloji naredba **goto** kojom se može „skočiti" na obeleženu naredbu.

Specijalnost *Geussa* je visestruko pndodeljivanje, kao u ovoj naredbi:

```
{ vr, vi } = eigrg(x),
```

Funkcija **eigrg** izračunava sopstvene vrednosti opšte matrice **x** i vraća dva veklora, **vr** tipa **Nx1** i **vi** tipa **Nx1** koji sadrze realne i imaginarne delove ih sopsivenih vrednosti.

Matrice zadajemo direktno, pndodaljivanjem svih njenih vrednosti nekoj promenljivoj. Za to se koriste vitičaste zagrade, a u programu sa do pojedinih elemenala matrice dolazi navodenjem indeksa u uglastim zagradama. Na primer:

```
x = { 1 2 3, 3 7 5, 3 7 6, 8 9 5, 4 5 6 };
y = x[3,3]
z = x[1 2 4, 1 3]
```

x[2,3] vraca sadrzaj drugog reda i trećeg slupca matrice **x**. **x[1 3 5, 4 7]** vraća submatricu koja nastaje presekom redova 1, 3 i 5 sa kolonama 4 i 7. **x[,3]** vraca treci stubac matrice **x**. **x[2:5,,]** vraca submatricu koja sadrži redove pocev od trećeg do petog u matrici **x**. Indeksiranje matrice može se pojaviti i na desnoj strani znaka pndodaljivanja:

```
y[rv, cv] = x,
```

I izrazi **rv** i **cv** mogu biti bilo koji izrazi koji sadrže veklore i matrice. Elemenli u **rv** se treliraju kao indeksi redova, slično elemenli u **cv** se treliraju kao indeksi kolone. Vrednosti **rv=0** i **cv=0** označavaju sva redove odnosno kolone matrice.

Moguće je indeksirati i skupove matrica, tekstova. Junkcija i procedura, što je takođe pozajmljeno iz C-a.

TIPOVI PODATAKA

Strikino govoreci, postoje samo matrice i teksi, uz razne vrsle konstanli. Matrica je dvodimenzioni niz brojeva u dvostrukoj preciznosti, dužine osam bajtova. To je tzv. IEEE format, kojeg Intelovi koprocesori poznaju kao „long real". Matrica tipa 1×1 je skalar, to jesl broj u običnom smislu. Ie reči a matrice samo sa po jednim redom ili kolonom su veklori. Evo najvažnijih operacija nad matricama:

[]	– indeksiranje
=	– pndodaljivanja
|	– vertikalno slepljivanje (*concatenation*)
	– horizontalno slepljivanje
CON	– učitavanje elementa matrice sa tastature
CONS	– unos teksla sa tastature
DECLARE	– priprema matrice ili teksta za rad
LET	– delinicija matrica
LOAD	– učitavanje matrice u memoriju
READR	– učitavanje iz *Gauss* daloteke
SAVE	– upis matrice procedura i teksla na disk,
SAVEXTOD	– pretvaranje matrice u *Gauss* skup
STOF	– pretvaranje teksta u matricu
SUBMAT	– ekstrahovanje submatrice
WRITER	– upis podatake u *Gauss* skup

Tip teksi (*string*) je isti kao u drugim programskim jezicima. Dodatno teksl se u memoriji završava nulom, tako da se može čitati direktno iz C programa.

Najvažnije radnje sa tekstom su

$+	– prisajedinjavanje lekslova
	– leksi se interpretira kao ime promenljive a ne kao lekst sam za sebe
CHRS	– pretvaranje vektora ASCII kodova u lekst
FTOCV	– tekstuelna reprezentacija broja u matrici tipa NxK
FTOS	– tekstuelna reprezentacija broja u matrici tipa 1×1
GETF	– učitavanje binarne ili ASCII datoteke u tekstuelnu promenljivu
INDCV	– nalazi indeks elementa u vektoru slova
LOWER	– pretvaranje u mala slova,
STOF	– pretvaranje teksta u broj
STRINDX	– nalazi početak jednog teksta u drugom tekstu
STRLEN	– dužina leksta
STRSECT	– izdvajanje leksta
UPPER	– pretvaranje u velika slova
VALS	– pretvaranje leksta u vaktor numeričkih ASCII kodova

Matrica može sadrzati i tekstove i tada se ispred njenog imena mora naci $ na primer sledeca naredba štampa celu matricu leksotova

print $x;

Obrnuta kosa crta (*backslash*) u tekstu označava upravljački simodl Tako je „\b" isto što i ASCII 8 (*backspace*) „\e" je isto što i ASCJJ 27 Jid

Strelica uvis ^ je simodl zamene U sledećoj naredbi

create f1 = olsdat with x,4,2,

Gauss će interpretirati **olsdat** kao ima datoteka koju treba učitati Tekođe će interpretirati **x** kao prefiks za imena **x1 x2 x3** i **x4** Ako želimo da do imena datoteke dodemo iz tekaluel·ne promenljiva koristimo simbole zamene

dalaset = „olsdat";
creale f1 = ^dataset with x,4,2;

Matrica takstova su naravno zgodne zbog štampanja numerickih matrica

Gauss se zasniva na radu sa matematičkim koprocesorom pa ima i posabne indikatore njegovih stanja NAN znači da se u loku rada koprocesora pojavilo beskonecno kao argument INF je lakođe signal beskonačnog ali koje je nastalo „prelivanjem" sadrzaja koprocesora (broj je preveliki za posloječi hardver) DEN znači da se pojavio prameli broj a ako takav broj ipak upotrebimo u proračunu – javlja se greška tipa UNN

Neke naredba imaju smisla samo lokom prevodanja Na primer **declara** objavljuje lip podatka na početku programa dok **external** obevešlava da neka procedura nije u sastavu programa koji se pravodi nego da je negde drugde na disku

Uvak postoji glavni program Naredbema #INCLUDE se – isto kso u C u – mogu automatski učitavali delovi programa sa diska Tok programa se menja naredoam **goto** ili strukturnenim naredbama DO WHILE ENDDO i DO UNTIL ENDDO Opšti oblik naredbe IF je IF ELSEIF ELSE ENDIF Konačno postoje funkcije i procedure u obliku potprograma a – kao u bejziku – posloji i funkcija od samo jednog reda na primer

fn area(r) = pi * r * r,

ARITMETIČKE OPERACIJE

Uobičajene aritmetičke operacije u *Gaussu* primaju za argumente i veklore i matrice Tako naredba

x = y * z

znači isto – množenje – i za brojeve i za matrice s tim što je krajnji elekal naravno drastično različit Operator deljenja / sa matricnim argumentima znaci rešavanje sistema linearnih jednačina Posloji i operacija / kojom se dve matrice dele element po elementi na primer prvi element prvog reda jedne matrice deli sa prvim elementom prvog reda druge matrice itd Slično je množenja matrica ali element po element Posloji i ^ što je dizanja na stepen element po elementi Ljubitelji lenzora i dovače se Kronekerovom proizvodu matrica koji se oznacava sa :*

y = x .*. z;

Rezultat ove operacije je matrica u kojoj je svaki element iz **x** skalarno pomnožen matricom u **z** Posloje još i operalori za horizontalni proizvod transponovanje vertikalno i horizontalno prisajedinjavanje (*concalenation*) matrica

Relacioni operatori posloje u dve varijante kao poređenje celih matrica ili kao poređenje matrica element po element Rezultat poređenja matrica je neka nova matrica koja sadrži samo 0 ili 1 na odgovarajucim mestima

Slično pel logičkih operalora – NOT AND OR XOR i EOV – mogu poredili direkino ili element po elementi Za razliku od relacionih rezultat njihove primene je uvek samo jedna logička vrednost TRUE ili FALSE

PROCEDURE I FUNKCIJE

Procedure se sastoje iz pet delova

PROC	– deklaracija procedure
LOCAL	– deklaracija lokalnih promenljivih jelo procedure
RETP	– povralak iz procedure
ENDP	– fizički kraj procedure

Evo procedure koja ima dve matrice kao argumente i vraca jednu matricu kso rezultat

```
PROC novaprocedura(x,y),
     LOCAL a, b,
     a = sqrt(x),
     b = y a^x,
     RETP (a * inv(b)'x),
ENDP;
```

Iz programa je možemo pozivati ovako

t = novaprocedura(z,u)

I procedure mogu biti ulazni argumenti drugih procedure

Jedna procedura može vratiti do 31 izlazni rezultat Broj vraćenih parametare navodi se kao argumeni sluzbane reci PROC Evo procedure za inverziju kompleksnog argumanta

```
proc(2) = cminv(xr, xi),
        local ixy, zr, zi,
        ixy = inv(xr) * xi,
        zr = inv(xr + xi * ixy),
        zi = -ixy * zr,
        retp (zr, zi);
endp;
```

U programu je zovemo ovako

{ zr, zi } = cminv(xr, xi)

Procedura može zvali neke druge programske jezike Sledeci primer poziva Baselovu funkciju iz *Microsoft C 5 1*, da bi izračunala Besselovu funkciju prvog reda

```
proc besselJ(n,x);
  local len, kind, bessel;
  len = rows(x)*cols(x),
  if rows(n) == 1, /* prvi red */
     kind = 1;
  else;
     if rowa(n)*cols(n) /= len;
        arrorlog „GRESKA – matrice nesaglasne',
     end,
  endif;
  kind = 10,
endif,
bessel zeros(2400 1),
loadexe bessel = bessel.gxe,
callexe /R bessel(x kind n len)
retp(x),
endp,
```

Promenljive n i x su vektori tipa Nx1 gde je N red Beselove funkcije Naredba if provera va (prvi red) da li020ulazni podaci imaju smisla Funkcija **zeros** formira matricu sastavljenu od nula tako da je u promenljivoj **bessel** sadržan vektor duzine 2400 Naredba **loadexe** učitava naredbu „stranog jezika iz datoteke **besseLgxe** koje se zatim sa **callexe** izvrsava

BRZA I FINA GRAFIKA

Grafika se u *Gaussu* deli na dve grupe naredbi – *Quick Graphics* („prirucna grafika") i *Publication Quality Graphics* (grafika dovoljno dobrog kvaliteta da može direktno uci u finalni leksl nekog dokumenta) Ova dva paketa se međusobno isključuju

"Brza grafika sadrzi osnovne i adnje crtanja stubica i raznih vrsta histograma uz različiJe vrsle normalnog i logaritmovanog grafika matematičkih funkcija Nema naredbi za štampanje mada se u prirucniku navodi kako se programski može inicirati ekvivalent naredbe **PrintScreen** sa tastature Ako je potrebna slika jedino je možemo „uhvatiti" nekim za to specijalizovanim programom Izgledom nacrtanog upravljamo prako globalnih konslanti (valičina slova vrsta prekida linije itd)

"Brza grafika" podrzava i prozore

"Kvalitetna grafika" je skup poziva grafičkih funkcija iz pakela *GraphiC*, kojeg je firma „Aptech" licencirala. Za izvršavanje ovog paketa neophodno je i svih 640K centralne memorije Rezolucija je 4096×3120 piksela što odgovara gustini od 455 piksela po inču stampanog dokumenta To je naravno više nego što se može prikazati na običnim laserskim štampači ma i ekranima pa se pre crtanja vrednosti zaokruzuju

Podržane su uodičajene grafičke kartice CGA EGA Hercules AT&T Osim raznih štampača podržani su i crtači od „Epsona" Houstona" i HP Gralici mogu biti sa stubičima konturni tipa histograma (procentnog i frekventnog) sa logaritamskom proporcijom po osama u polarnim koordinalama Možemo crtati i 3D površi sa uklanjanjem skrivenih linija Postoje četiri fonta sa i bez serifa grčka slova sa matematičkim simodlima i masne slova

"Kvalitetnu" grafiku možemo crtati interaktivno ili iz programa U prvom slučaju postoji i zumiranje Globalnih parametara ima nekoliko desetina Crtanje traja duže ali je besprekornog kvaliteta čak i na matričnom stampaču

NEMA MU RAVNA

Po sveoduhvatnosti metematičkih020operacija *Gaussu* nema ravna medu jazicima. Lak je za učenje Nije za svakoga ali ako se bavite numerikom u bilo kojem obliku – alternativa *Gaussu* su samo mučni prevodioci za FOR TRAN

Korisna adresa

Aptech Systems Inc
26250 196th Place South East
Kent Washinglon 98042, USA
Tel (206) 6679 Fax (206) 630-1220

KOD IZ ĐAVOLJEG SEMENA

U nastavku serije članaka o kriptografskim sistemima upoznacemo DES, američki standard za šifriranje podataka. Ukratko, napravljen je u IBM-u – i ne mogu ga razbiti ni genijalci.

Admir Abdurahmanović

Pnča o DES-u nlja baš jednostavne I kao što cete se uvjeriti nije još došla do kraja. Početak je znan – DES je razvijen iz đavoljeg sjemena (kripto sistem LUCIFER) u a gdje bi drugdje IBM u Kada je stvar napravljena niko nije znao zasto to radi osim evenlualno (ili mo zda vjerovalno) autora Prije priznavanja kao zvaničnog standarda (1977) a da bi se provje rila „snaga" algoritma potrošene su skoro dvije ljudske decenije istrazivanja najeminentnijih strucnjaka u pokusaju da se DES razbije Niko nije uspio

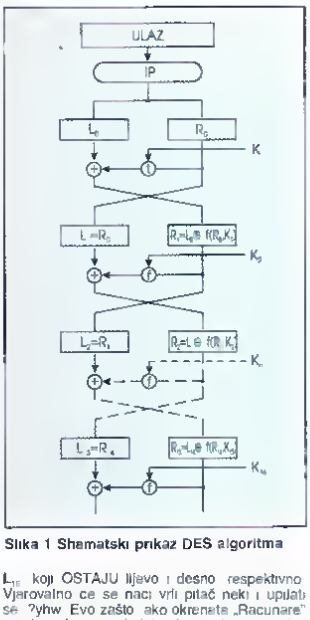
U lerminologiji kriptografije DES je kompozicija više meloda šifriranja (*product cipher*) Ima jedan kljuc za desifriranje (*one key system*) i openra nad blokovima od 64 bita (*block-cipher*)

KAKO RADI

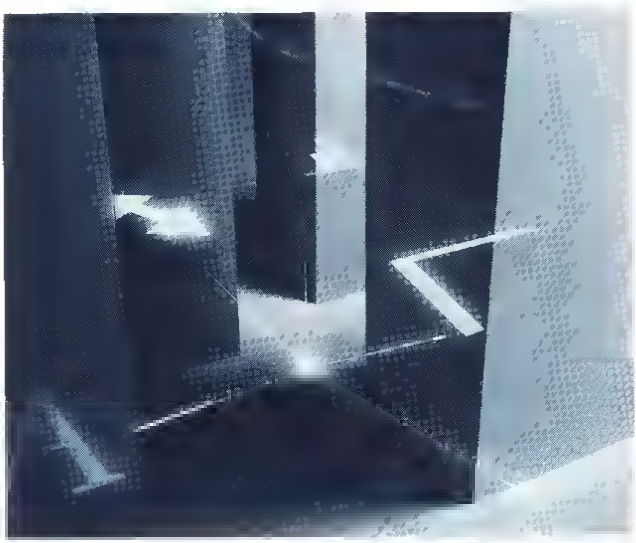
Za sifriranje DES om su potrebne dvije stvari – kljuc **K** od 56 bita i tajna poruka **M** od n×64 bita (Za potrebe objašnjenja smatracemo da je n=1.) U opcem slučaju, dužina poruke (u bitovima) ne mora biti djeljiva sa 64 – tada se na kraj dodaju nula bitovi (*padding*)) Algori tam nije jednostavno opisati u vidu jednog dija grama toka, pa se najčešce objašnjava pomo ću sheme koje se lako razumijevaju

Ključ **K** se obično isporuči kao 8 bajtova pa se najviši bitovi ispusta ili se upotrijeba kao bitovi *pantete* Prije upotrebe ključa isli se u procaduri muckanja ključica (*key schedule*) nasilno isjecka (na bitove), razbaruši se (permutacija *permuted choice* 1) podijeli na dvija „ls Je" polovine C i D Je se zatim miješa sve na istu stranu (šiftovanje dala je matrica koja de finira koliko se šifta) i uzima se po 48 bita koji se opet malko razbaruše (*permuted choice* 2) Sve se krčka toliko da se dobije 16 kljucica (po 48 bita) – za svaku iteraciju de/sifriranja po je dan. Interesantno je da su C_0/D_0 i C_{16}/D_{16} jednaki Autor DES-a su „namjestili" da se ključići mogu dobijati i obrnutim redoslijedom što je korisno kada se neka poruka desifrira Također se može desiti da neki ključevi generiraju iste ključiće što na treba da zabrinjava jer takvih ključeva (*weak keys*) ima 4 a onih koji generiraju samo dva različita kljucica (*semiweak keys*) ima 6, što čini 10 od 2^{56} mogucih kljuce va

Slijedi objašnjenje sheme 1 Gledajuci odo zgo nadole najprije primjećujemo inicijalnu permutaciju **IP** koja samo razmjesti bilove po ruke **M** Inverzna ovoj je permutacija FP (na dnu sheme) $\mathbf{L_0}$ i $\mathbf{R_0}$ koji su lijevih odnosno de snih 32 bita od onoga šta se dobije nakon IP $\mathbf{K_i}$ su ključici koji se dobijaju po već izloženom receptu + je bitni XOR *f* je funkcija dva ar gumenta $\mathbf{R_i}$ i $\mathbf{K_{i+1}}$ za i=0 1 ...15 Rezultat fun kcije *f* se kao što se vidi XORuje sa odgovarajućim L_i i to postaje novi R_{i+1}, a stari R_i posta je L_{i+1} To ide tako sve dok se ne dobiju R_{16} i



Slika 1 Shematski prikaz DES algoritma



$\mathbf{L_{16}}$ koji OSTAJU lijevo i desno respektivno Vjerovalno ce se naci vrli pitač neki i upitati se ?yhw Evo zašto ako okrenete „Racunare" „naglavacke i pogledate shemu 1 primijetićete da DES i DES naglavačke s-ice Zapravo ličili bi poprilično ako bi se Io još zaokrenulo po vertikalnoj osi Upravo tako radi desifriranje u DES-u Pošlo ovo objašnjenje nije najpreci znija avo naslo ubjedljivije argumentacije Na me za i=1,2 ... 16 vnjede jednakostr

$$L_i = R_{i-1}$$
$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$$

Znači, ako su svi ključići dostupni tada je i ta iteracija šifriranje potpuno odredena sa i 1 vom iteracijom No, zbog osobine operacije XOR da je sama sebi inverzna Io očito vnjede jednakosli

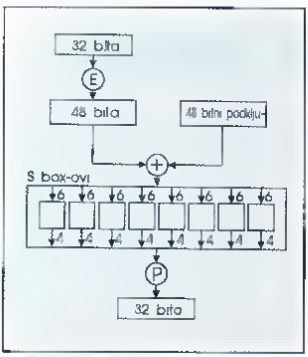
$$R_{i-1} = L_i$$
$$L_{i-1} = R_i \oplus f(R_{i-1}, K_i)$$

a posljednje je isto što i

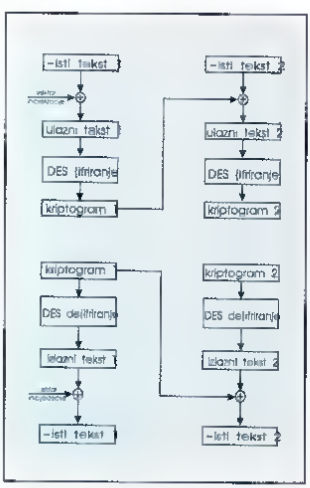
$$L_{i-1} = R_i \oplus f(L_i, K_i)$$

Ovim smo pokazali da se i-1 va iteracija (naime njen rezultat) može izrazili pomoću i te

Najpikantniji dio algoritma je funkcija *f* (shema 2) Argumenti su i-li R (32 bita) i i ~ 1 vi ključić (48 bita) Ova relativna razlika se prevazilazi tako šlo se R ekspandira sa transformacijom E To se zatim XORuje i dobijeni (me du)rezultat se dijali na 8 djelova po 6 bita Svaki dio se preslikava preko odgovarajuceg



Slika 2 Fukncija *f* **u DES algoritmu**



Slika 3 Shema DES CBS metoda

S-box a u svoju sliku. Pretpostavimo da je jedan 6-bitni dio označen sa $b_1b_2b_3b_4b_5b_6$. Svaki pojedinačni S box se predstavlja u obliku tabele te dimenzija 4 sa 16. Bitovi b_1b_6 te $b_2b_3b_4b_5$ određuju u koju **4-bitnu** vrijednost se preslikava polazna sekvenca bita. Dobro ste pročitali – **S box**-ovi su nelinearne (i nelnvertibilne) transformacije prostora 2^6 u 2^4. Upravo zbog pitanja koje se samo nameće: prelazimo na sljedeći pasus.

ZAŠTO RADI?

Odgovor možete (zasada) dobiti samo na dva načina. Prvo da se obratite kompaniji IBM sa molbom da vam pošalju detaljno objašnjenje. Drugo da se obratite izvjesnoj kompaniji (izdavačkoj kući) na Sinjusu 5 sa molbom da vam pošalju najnovije izdanje svog best sellera u kojem možete potražiti odgovor.

Ono što možemo dati umjesto odgovora svodi se na diskusiju osobina ovog algoritma. Premda je DES algoritam javno publikovan, izvjesni krilenji dizajna se drže tajnima. neročito kriterij za odabir **S-box**-ova. P permutacije i *key schedule* procedure. Na primjer, pokazano je da se pogodnim odabirom vrijednosti **S-box**-ova (što nije dokazano da vrijedi za DES ove **S-box** ova) može smanjiti broj potrebnih ključeva za pretragu maloclom grube sile na „samo" 25% od onih koliko (se smatra da) treba da bi se razbio DES. Neki su kriticari nalazili da **S-box** ovi imaju „čudne" osobine (u izvještaju stoji riječ „peculiar").

Svaki DES ov **S-box** ima osobinu da promjena samo jednog bita u argumentu ima za posljedicu promjenu bar dva bita rezultata. Ovo je važno jer rezultira propagiranjem greške (*error propagation*) i jedan od konsnih aspekata ove osobine će biti jasan kasnije u dijelu koji objašnjava DES CBC mod de/šifriranja.

Općenito, jedna od vrlo poželjnih osobina kripto sistema je efekat lavine (*avelanche ef fect*) koji se sastoji u lome da „mala" promjena ulaznih podataka uzrokuje veliku promjenu u izlaznim rezultatima. Specijalno za DES je pokazano da poslije pet iteracija svaki od 64 bita (posmatrajući odgovarajuće L i R zajedno) zavisi od svih bitova poruke i svih bitova ključa. Kada se DES „osakati" tako da ima samo dvije iteracije (umjesto 16), može se razbiti uočavanjem korelacije između ulazne poruke i L_2 i R_2. Korelacija se (nije dokazano) gubi nakon 8 iteracija. Da bi se osigurao što bolji efekat lavine preporučavano je da se poveca broj iteracije sa 16 na 32.

Interesantna osobina DES-a je ta da je rezultat šifriranja poruke **M** ključem **K** jednak rezultatu šifriranja **M** ključem koji je bitni komplement ključa **K**, pa se dobijeno bitno komplementira. Ova osobina nije odveć poželjna jer u slučaju razbijanja metodom izabrane poruke (*chosen plaintext* attack) treba pretražiti „samo" 2^{55} umjesto 2^{56} ključeva. Ugled DES-a je dosada najviše uzdrman prošle jeseni kada je objavljeno da Adi Šamir (Shamir) izraelski naucnik, metodom izabrane poruke može da razbije DES. To jest da otkrije ključ. Da podsjetimo još jednom, ovo znaci da razbijač mora da bude u stanju da obezbijedi (bar) dio nake poruke i njen kriptogram. (Isline radi, za spomenuti razultat se navodi da njemu treba „pažljivo konstruirani cisti tekst".) Vec smo napomenuli da se osakakačeni DES mogao razbiti eli Šamirov razultat odnosi se na svih 16 iteracija, dakle na *pravi DES standard*.

DES CBC MOD

U praksi je moguce da se obezbijede uslovi za razbijanje DES a metodom izabrane poruke. To se naročito odnosi na DES ECB (*elec Ironic code book*) mod u kojem se ulazna poruka podijeli na blokove od po 64 bita i svaki blok se šifrira *istim* ključem. Generalno se ne preporuca da se ulazna poruka podijeli na blokove i šifrira istim ključem. Na primjer, pretpostavimo da se DES primjenjuje u nekoj banci i da se šifriraju slogovi svakog konsnika i to svako polje posebno. Iako da imamo slucaj da se (A 10000) (B 12345) i (C, 11111) šifriranjem pretvaraju u (X1 Y1) (X2 Y2) i (X3 Y3). Ij pa rova koji nem više ništa ne govore. Ali mozda neko moze da zamjeni sadržaje Y1 i Y3, bez obzire na to sto na zna ključ kojim bi razbio sistem. Što je ekvivalentno razbijenju sistema po sleti koja se može prouzrokovati. Dalja razbijac može da dozna da par (X1 Y1) predstavlja šifrirani oblik za (A, 10000). Sa dovoljno mnogo ovakvih podalaka sa dovoljno mnogo znanja u razbijanju DES a i sa dovoljno mnogo racunarskih resursa, razbijac moze da se nada uspjehu. Čak i da razbijač koristi metodu grube sile, ako bi bio u mogucnosti de obezbijedi n parova parova ((x y))X Y)) moza da prosjeći no vrijeme pretrage umanji za faktor n. kon steci paralelno n računara.

Ova dva primjera su dovoljna de ilustriraju da se DES uglavnom koristi samo kao polazni algoritam za neki složeniji koji nema pomenute nedostatke. Jedan takav je DES CBC (*cipher block chaining*) metod predstavljen shemom 3. Osim DES CBC moda postoje još CFB (*cipher feedback*) CBCP (*cipher block chaining with plaintext feadback*) i CFBV (*stream cipher feedback with vector feedback*) modovi koji će prirodno biti ostavljani za neku tracu priliku. Šifriranjem prvog bloka od 64 bita sa ključem **K** dobija se **C**., drugi blok se šifrira sa ključem **K XOR C**, i tako redom. Ovim metodom je obezbjedeno da jedan šifrirani blok ne ovisi samo od ključa **K** i sadržaje bloka cistog teksta već i od prethodnog bloka, a ovaj opet od svog prethodnika i tako unazad do prvog. Sada se ne može govoriti o mogucnosti da neki uljez moza da ubaci neki svoj blok umjesto nekog originalnog jer time stvara grešku prilikom da šifriranja. Ovaj metod ima jos jednu dobru osobinu. Pretpostavimo da je uslijed djelovanja uljeze ili zbog loših komunikacionih uvjeta došlo do greške na nekom bloku. Sasvim očekivano greška će se pojaviti u procesu dešifriranja ali greška neće uništiti ostatak poruke! Dva bloka „iza" greške je moguce nastaviti proces dasifriranja jer se uzme blok (za kojeg se pretpostavi da je O.K.) XORuje se sa ključem **K** i pokuša se deširirali naredni blok. Ako se dobiju smisleni podaci. To znači da je dio sa greškom „preskočen".

MAC

Kazali smo „smisleni podaci" a smisao se na „zna uvijek. Na primjer u rezultatu nekog mjerenja „lačan" podatak 32 67443903678 bude zamjenjen sa „nelačnim" 23 55679330769. Kako da znamo šta je lačno? Rješenje dilame nalazimo konštenjem tzv. MAC kodova (*message authentication code*). Za ovu priliku cemo samo reći da su ovo MAC kodovi dodaju na poruke za koje nam je vazno da prilikom prenosa podalake ostaju u originalnom stanju do u bi. Kako možemo biti sigurni da prilikom prenosa nija doslo do greške? Jasno da na možemo biti 100% sigurni al 99,99% možemo ako izvedemo pogodno odabranu transformaciju ned porukom koja ima osobinu koju smo vec neveli – propagiranja greške.

Postoji vise takvih trasformacija a nama je uspul da pomenamo MAC baziran na DES u. U osnovi „odradi se" DES CBC tako se zapamte same prva 4 bajta zadnjeg bloka koji je dobijen šifriranjem. Ako se poslije prenosa i deši fnirania poruke ne može doći isih 4 bajta on da imamo razloga da se zabrinemo, a ako se dobiju isti bajtovi onda mozemo reci sebi da nije moguce da budemo lako malerozni da greška daje isti kod kao i original.

Uprkos svim rečenim zamjerkama DES se i dalje koristi u bankarskim transakcijama što je znak da se još uvijek smatra pouzdanim. Takoder još uvijek se daje zabrana na izvoz DES programa van USA (mada ima primjera koji govore da se nekima gleda kroz prste). Razbijanje DES CBC (i drugih) modova je dalako kompleksniji problem od „akadamskog" razbijanja DES a. Možda je „najobjeklivnije" reci da postoje algoritmi koji su manja diskutabilni koji se „moderniji" koji su upotrebljiviji koji ni/su standardizirani koji su sporiji koji ni/su patentirani. Cijela ova priča nije išla samo u prazno. Izvorne i izvršne programe za DES provjeru ključa kao i MAC možete da dodijete na Sezamu.

UMETNOST MODELIRANJA

Pažljivijl čitalac „Računara" primetio je, verovatno, jednu od bitnih karakteristika geneze pisanja o geometrijskom modeliranju. Iz članka u članak predstavljane su razne metode sa uvek istim zaključkom – dobro ali ima bolje. Ovoga puta ćemo stati i reći: ovo je najbolja alatka koja se trenutno koristi u geometrijskom modeliranju objekata.

Slavko Korać i Milan Bojanović

Reč je o tzv. NeUniformnim Racionelnim Splajnovima, alias **NURBS**, u koje se zaklinju svi projektanti brodova, aviona i automobila. Objasnićemo mane i vrline ovog trojanskog konja u tehnologiji projektovanja pomoću računara, uz dužnu napomenu da ovaj put nećemo koristiti ješku matematičku artiljeriju. Svi neophodni fundamenti ovih metoda objašnjeni su u prethodnim brojevima *Računara*.

ZAŠTO NURBS

U sedamdesetim godinama projektanti su prepoznali jednu vrlo važnu klasu problema u svom svakodnevnom poslu – kako naći jedinstven metod kojim će se pokriti najrazličitije familije krivih linija i površi. Precizuje trebalo je obezbediti matematički alat koji će, na primer, jednim tipom krive linije pokriti i pravu liniju i konusni presek i krivu proizvoljnog oblika! Bitno je istaći da se (ranije pomenutim) Bezierovim ili B-splajn metodama to moglo postići samo aproksimativno.

U cilju rešenja navedenih problema, niz istaknutih pojedinaca na univerzitetima (osobito University of Utah) i istraživačkim centrima moćnih kompanija (na primer „Boeing" SDRC i drugi) došli su do zaključka da se racionalnim formama neuniformnih B-splajnova mogu uspešno ispuniti sledeća očekivanja projektanata:

- *NURBS*-ovi pružaju jedinstvenu formu za predstavljanje i projektovanje kako standardnih analitičkih formi (konusnih preseka, cilindričnih površi i sličnо) tako i nejrazličitijih oblika krivih i površi.
- pomeranjem tačaka kontrolnog poligona menja se oblik krive koja je definisana njime (detaljno je bilo reči o ovome u prošlom broju *Računara*), međutim, *NURBS*-ovi omogućavaju i promenu oblika preko tzv. težinskih koeficijenata. Što je još jedan doprinos u fleksibilnosti promene oblika krive linije ili površi.
- Sa aspekta numeričke matematike, primenjivane numeričke šeme su stabilne.
- *NURBS*-ovi imaju jasnu geometrijsku interpretaciju i lako ih prihvalaju projektanti koji imaju iskustva sa klasičnom nacrtnom geometrijom,
- NURBS-ovi su vrlo pogodni sa aspekta računarske grafike. Invarijantni su u odnosu na skaliranje, rotaciju i translaciju.

Međutim, da li je dovoljno samo ispuniti zahteve krajnjih korisnika? Odgovor je svakeko negativan osobito ako ge deju stručnjaci koji treba da naprave odgovarajuće softversko rešenje i implementiraju ga na računar. Slutite da je vec reč o negativnim stranama *NURBS*-ova. Potrebni su izuzetno veliki hardverski resursi (memorijski kapaciteti) za definisanje tradicionalnih krivih linija i površi. Na primer, da bi se predstavila kružnica preko *NURBS*-ova neophodno je definisati sedam kontrolnih tačaka i 10 čvorova. Tradicionalni postupak zahteva centar kružnice, radius i vektor normalan na raven u kojoj leži kružnica. Za kružnicu u 3D prostoru potrebno je 38 podataka a kod klasičnog predstavljanja svega 7. Ovde leži i odgovor zašto *NURBS*-ova skoro da nema u CAD paketima koji se baziraju na PC platformama.

Osim ovih problema ponekad su prisutni numerički „kolapsi" pri računanju preseka površ-površ i slično. Međutim, ne smemo zaboraviti da sve metode iz ove oblasti imaju svoje čorsokake.

ŠTA JE NURBS?

Uz još jednu ogradu da ce matematički aparat biti sveden na minimum, da bude što bezbolnije za one koji ne simpatišu kompliko vana dokazivanja, predstavićemo šta su to u sušini *NURBS*-ovi. Podimo pre svega od geometrijske interpretacije homogenih koordinata koje su nezamenjiva alatka u računarskoj grafici.

Označimo ose u Euklidovom 3D prostoru sa X, Y i W, a sa x,y drugi koordinatni sistem gde je x paralelno X, y paralelno Y i neka koordinatni početak leži u (X,Y,W)=(0,0,1) (slika 1). Neka bilo koja tačka P' u projektivnoj ravni W=1 definiše liniju OP. Linije OP može biti definisana sa bilo kojom tačkom koja leži na ovoj liniji. Koordinate (XP,YP,WP) tacke P su *homogene koordinate* P. Očigledno da se pozicija P duž linije OP' direktno određuje položajem te tačke u odnosu na tačku O. Tako, na primer, ako su P i Q dva različite tačke duž pravca OP', onda su njihove koordinate homogene koordinate tačke P'. Drugim rečima

(XP,YP,WP)

i

(XQ,YQ,WQ)=(uXP,uYP,uWP), $u \neq 0$

su homogene koordinate iste tačke.

Razmotrimo sada dva specijalna slučaja. Prvi kada je P=P'. U tom slučaju je P=(x,y,1) i koordinate se dobijaju iz jednostavnih relacija x=X/W, y=Y/W. Drugi slučaj, ako R leži u ravni X,Y onde OR ne preseca ravan x,y. Odgovarajuća projekcija R se predstavlja kao tačka u beskonačnosti. Napokon, kakva je veze gore pomenutih koordineta se *NURBS* ovima? Pozajmimo gornju šemu i prevedimo je na domen krivih u 3D prostoru. Dakle, dodajemo još jednu koordinatu z i u tom slučaju tačka iz Euklidovog prostora (x,y,z) ima homogene koordinate (x,y,z,1) koje su dobijene prevođenjem iz tačka iz kordinetnog sistem (Wx,Wy,Wz,W). Ovaj postupak možemo opisati na sledeći način

$$H(Wx,Wy,Wz,W)=\begin{cases} Wx/W, Wy/W, Wz/W \text{ za } W\neq 0 \\ \text{tačke u beskonačnosti, na pravcu koji spaja koordinatni početak } W=0 \\ (x,y,z) \text{ za } W=0 \end{cases}$$

Na osnovu modela prevođenja Jačaka iz tzv 4D u 3D prostor Tiler (Tiller) je 1983. godine uspostavio sličnu analogiju za B-spline krive. Kontrolne tačke Pi koje definišu ne racionalne B-spline, mogu se shodno gornjoj semi transformisati i na taj nacin dobiti obrazac za računanje *NURBS*-a kao

$$C(t)=\frac{\left(\sum_{i=0}^{n} N_{i,p}(t) W_i P_i\right)}{\left(\sum_{i=0}^{n} N_{i,p}(t) W_i\right)}=\sum_{i=0}^{n} R_i(t) P_i$$

gde $R_{i,p}(t)$ – predstavlje racionalne bazne funkcije, a $N_{i,p}(t)$ bazne funkcije ne-uniformnog B splajna.

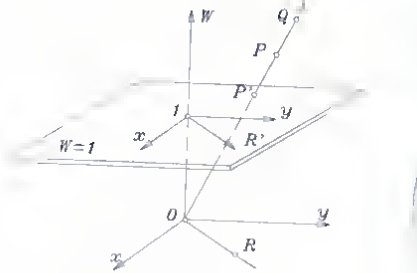
U gornjem izrazu figuriše ranije pomenuti parametar W. On predstavlja tzv. težinski koeficijent kojim se osim pomeranjem temena kontrolnog poligona može efikasno i lokalno menjati oblik krive. Sve ovo se može dokazati čisto matematički ili pak pokazati analizom ponašanja racionalnih baznih funkcija.

Najvemiji ljubitelji *NURBS*-ova su kao što smo rekli oni koji u svojim projektima barataju sa vrlo složenim površima. Zašto baš oni? Pokušaćemo da Jo objasnimo na primeru projekJanja koji se bavi dizajniranjem karoserije automobila. Pretpostavimo da dizajner želi na zakrivljenoj haubi motora da napravi malo zaravnjenje kako bi prićvrstio, na primer, znak merke automobila. Stare matematičke alatke (ovo znači iz 60-ih ili 70-ih godina) nisu omogucavale lokalnu kontrolu oblika pa bi naš projektant promenom geometrije jednog dela površi izazvao degeneraciju oblika ostalog dela haube. Normalno i ova degeneracija se može kontrolisati uvođenjem određenih ograničenja, ali nije to cilj. Cilj je postignut ako je projektant u stanju da menja oblik krive ili površi samo na onom mestu na kome on to želi. To upravo omogućavaju racionalni postupci.

Ovde smo namerno uopštili pojam i rekli racionalni postupci jer racionalne funkcije se ne izvode samo iz (ne)uniformnih B-splajnova, već se to može uraditi i sa Bezierovim polinomima. Ovaj poslednji postupak je osob to koristan kada se govori o predstavljanju konusnih preseka preko racionalnih Bezierovih polinoma.

KONUSNI PRESECI

Još je Liming 1944. godine uočio da konusni preseci predstavljaju efikasno sredstvo za opisivanje poprecnih preseka trupa aviona pre svega jer je njima moguce obuhvatiti različite krive drugog reda (hiperbola, parabola, elipsa). Ovaj pristup je osobito pogodan za opisivanje trupova vazduhoplova vrlo glatkih kontura i sa poslojanjem jedne linije maksimalne sinne (na primer jednice), gde se poprečni presek trupa može predstaviti na osnovu dva konusna preseka (polazi se od toga da trup ima raven simetrije). Preseci složenijeg oblika mogu se do-



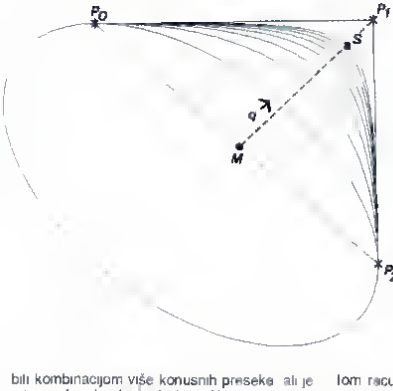
Slika 1 Geometrijska interpretacija homogenih koordinata



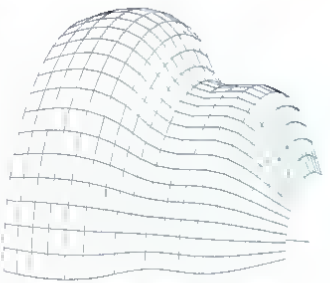
Slika 3 Deo trupa letelice uradene metodom konusnih preseka



Slika 5 Površ dobijena zadevanjem krive linije i vektora po kome se ona translira



Slika 2 Familija konusnih preseka dobijena postupkom *cross-section design*



Slika 4 Površ nastala rotacijom *NURBS*-a

bili kombinacijom više konusnih preseka ali je ovaj postupak vrlo često komplikovan

U nastavku ce ukratko biti opisan model predstavljanja parametrizíranog konusnog preseka preko racionalnih kvadratnih funkcija e koji je korišcen kao osnovni model u softveru za oblikovanje trupova vazduhoplova razvijen na katedri za vazhuhoplovno mašinsvo Mašinskog fakullela u Beogradu Konusni presek se može predstaviti preko racionalne forme Bezierovog polinoma kao

$$c(t) = \frac{B_{0,2}(t)W_0P_0 + B_{1,2}(t)W_1P_1 + B_{2,2}(t)W_2P_2}{B_{0,2}(t)W_0 + B_{1,2}(t)W_1 + B_{2,2}(t)W_2}$$

gde parametar $t \in [2\ 1]$ e sa B_i su predstavljene Bezierove neracionalna funkcije

$B_{0,2} = (1-t)^2$
$B_{1,2} = 2t(1-t)$
$b_{1,3} = t^2$

Izbegavanjem za ovu priliku nepolrebnih dokazivanja može se pokazati da se tezinski faktori u jednačini konusnog preseka mogu odrediti na sledeći način

Fiksiranjem vrednosti $W_0 = W_2 = 1$ nado slajuci težinski faklor W_1 se određuje iz izraza

$W_1 = q/(1-q)$

Parametar *q* koji predstavlja vrlo pogodno sredstvo za promenu oblika konusnog preseka moze se predstaviti na različite nacine na primer

$q = MS / (MS + SP_1)$

Metod interaklivnog pomeranja lemene lačke S konusnog preseka (digitalizacijom) a polom racunanja parametra *q* Iz koge se neposredno dobija vrednosl tezinskog koeficijenta W_1 korišlen je u softveru za interaklivno oblikovanje vazduhoplova Kontinuelnom promenom parametra *q* od preseka do preseka dobijaju se glatka površi objekata koji se modeliraju Ovaj postupak je poznat pod nazivom *cross section design* a familija konusnih preseka dobijene po gornjem obrascu date je na slici 2 Po ovom principu mogu se uradili i trupovi camaca delova karoserija automobila i slično Na slici 3 je prikazan deo trupa lalelice koja je uređena metodom konusnih preseka a koji su predstavljeni preko racionalne forme Bezierovog polinoma drugog stepena Kompletan kod za generisanje konusnih preseka uskoro će se naći i na Sazamu

KAKO GA UPOTREBITI

Istakli smo jednu od bitnijh oblasti za efikasnu primenu NURBS-ova – definisanje tzv okvašenih površi vazduhoplova brodova ili automobila koji je u žargonu poznat kao *strako vanja* Predstavili smo u stvari postupak poznat pod nazivom *skinning* međutim da bismo definisali povrsi na osnovu krivih linija postoji još niz postupaka Na primer u *AutoCAD*-u postoji komanda REVSURF kojom se površ dobija ako se oko osa rotacije rotira kriva linija (napominjemo da *AutoCAD* ne koristi *NURBS*-ove za definisanje linije) za ugao koji se prethodno definiše Ovim postupkom se može dobiti površ (*surfaces of revolution* slika 4) nastala rotacijom *NURBS*-a i smatra se da su ovo najčešće upotrebljavani oblik površi u CAD u Ako se pak, površ formira rotacijom kriva linije, e čije tačke ne opisuju previine kružnice – onda je rec o tzv *swingig* tehnici e dobijene površi se nazivaju *swing* povrsi Kad smo vec pomenuli *AutoCAD* pomenimo *ruled surfaces* i *tabulated surfaces* (slika 5) Za definiciju prvih je potrebno i dovoljno definisati dva krive linije u prostoru dok se druge formiraju zadevanjem krive linije i vektora po kome se ona translira Zanimljivosli radi oblikovanje lelelonske slusalice spada sa stanovišla geometrijskog modeliranja u jeden od složenijih postupaka gde se koriste *free form extruded surfaces*.

Šta još reci u zaključku ovog teksta o NURBS ovima a da bude korisno za sadašnje ili potencijalne korisnika CAD softverskih paketa? Prvo i osnovno je da će vrlo brzo i vlasnici presonalnih racunara ne ekranima svojih ljubimaca projektovali na nacin koji je za sada privilegija onih koji rade na radnim stanicama Ovo je evidentno i po tome što su već komercijalno raspoložive biblioteke za pisanje sopstvanih CAD paketa baziranih na *NURBS*-ovima (jedan ovakav paket ce uskoro biti predstavljen i čitaocima *Računara*) Druga vrlo bitna informacija je da su *NURBS*-ovi postali i oficijelni deo IGES standarda. PHIGS+ gralicke biblioteka a u njegove moci su loliko uvereni projekianli softvera u „Hewlett-Packardu" da su razvili specijalni procesor koji u njihovim radnim stanicama ima samo zadalak NURBS *number crunching*

Dakle utvrdili smo za zašlo je Siradivanje va violina najbolja ostalo je jos samo da je kupimo i što je najbitnije da naučimo dobro sviramo

RADOST PROJEKTOVANJA

Retko imamo priliku da predstavimo domaći softverski proizvod koji može da se takmiči sa vrhunskim svetskim programskim paketima. *qCAD* je upravo takav paket, a namenjen je efikasnom projektovanju štampanih ploča u savremenoj elektronici. Zašto sam se, sa preko dvadeset godina rada na razvoju hardvera i bogatim iskustvom u svim tehnikama projektovanja i pripreme štampanih ploča, odlučio da već tri godine koristim *qCAD*?

Dragoslav Jovanović

Za razliku od sličnih programskih paketa qCAD – autorsko delo Aleksandra Čarapića – ne zahteva dane proučavanja uputstva za upotrebu kako bi program počeo da se koristi. Iako ima velike mogućnosti relativno je lak i jednostavan za rad – dovoljno je jedno popodne sedenja za računarom da bi se uspešno krenulo. Zato je uputstvo prilično kratko (pedesetak stranica) i uglavnom upućuje na programske *screen* ove i ugrađene *help*-ove. Pošto je namenjen pre svega profesionalcima ne bavi se osnovama iz oblasti projektovanja i pripreme štampanih ploča, kao ni opisom osnovnih termina, ali će se i prosečan poznavalac teme brzo uklopiti.

PRE NEGO ŠTO POČNETE

Minimalna hardverska konfiguracija potrebna za rad je zaista minimalna: IBM kompatibilan računar sa 640K RAM-a, flopi ili hard disk sa 720 K slobodnog prostora, instalirana sa 84, 101 ili 102 tastera i Hercules, CGA, EGA, VGA ili Super VGA grafička kartica sa monohromatskim monitorom i „Genius" ili „Logitech" kompatibilan miš. Za ozbiljniji rad je naravno poželjna što jača konfiguracija i pre svega kvalitetan printer ili ploter kao osnov za dobijanje kvalitetnih filmova za izradu štampane ploče.

Ceo paket je smešten na jednoj (!) disketi od 1.2 MB (zauzima oko 1 MB) što govori o efikasnosti programskog koda, posebno kada se imaju u vidu performanse – naročito *Autorouter*a. Sastoji se od devet programa, inicijalne biblioteke elemenata i pomoćne datoteke za definiciju hardverskog okruženja. Tih devet radnih programa su:

Import

program za konverziju „stiane" net-liste (lista povezivanja elemenata) iz drugih programa (*pCAD* i *ORCAD*) kao i konverziju *qCAD* net liste iz odgovarajuceg * PCB fajla u *pCAD* ili *ORCAD* formal

LIBEdit

program za editovanje odgovarajuće biblioteke elemenata (PACKAGE LIB) – definisanje geometrije elemenata i korespodentnih pedova

PCBEdit

program za glavno editovanje štampane ploče (fajl se ekstenzijom PCB) – definiše se štampane ploce pozivaju elementi iz biblioteke definisu signali na pinovima

HandRout

program koji se opciono koristi pre ili posle autorutera za manualna intervencije kod pojedinih signala

AutoRout

program koji predstavlja ubedljivo najjače oruđe *qCAD*-a i kako mu i ime kaže služi za puno 100% automatsko povezivanje do 6 slojeva na ploči sa mogućnom optimizacijom dužine veza, broja prolaznih rupa i kružnih prolaza

MetaMake

program za kreiranje potrebnih crteža na osnovu rada autorutera. Rezultat su odgovarajuci * MET fajlovi u vektorskoj grafici koji se po želji mogu i editovati ili se odštampati na štampacu ili ploteru

MetaEdit

program za editovanje grafičkih * MET fajlova (unošenje dodatnih oznaka ili tekstova na bilo kom sloju ili pojedinim slojevima štampane ploče kao i eventualno dotenjanje veza izmena veličine pedova debljine linija...)

MetaTest

Program za finalno testiranje svih * MET fajlova

MetaDraw

program za štampanje * MET fajlova (izlaz na matricne i laserske printere obične i foto-plotere...)

Instaliranje programa je krajnje jednostavno: disketa se stavi u drajv i pozove INSTALL.EXE. Definisanjem osnovnog hardverskog okruženja i direktorijuma u koji cemo smestiti paket ceo posao je automatsk za više

KAKO DO ŠTAMPANE PLOČE

Da bi se krenulo treba: naravno imati elektricnu shemu uređaja za koji kreiramo štampanu plocu. Ma koliko uređaj bio komplikovan *qCAD* se nece zbuniti ograničenja skoro da nema. Uglavnom ih diktira veličina raspoložive RAM memorije:

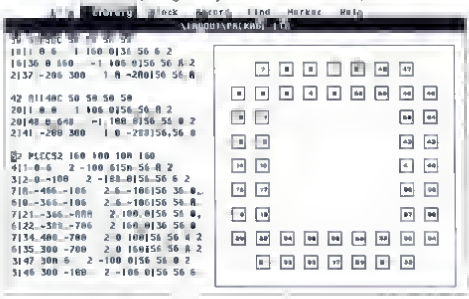
maksimalna veličina ploče	800 x 800 mm
maksimalan broj pinova	4000
maksimalan broj elemenata	1000
maksimalan broj signala	1000
maksimalan broj slojeva	6

Posebno naglašavamo da program nema nikakvih rasterskih ograničenja – sve operacije se mogu izvoditi do rezolucije od 1 mila (1/1000 inča)! Ovo je naročito značajno u radu sa elementima čiji se pinovi ne nalaze na standardnom rasteru (1/10 ili 1/20 inča) – RS 232 (Sub D) konektori na primer.

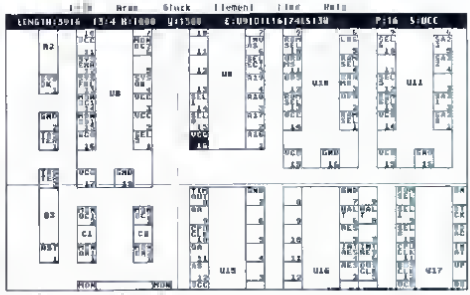
Ukoliko je shema regularno iscrtana u npr *ORCAD*-u sve što je za početak potrebno je da pozovemo IMPORT.EXE koji će raspoloziti vu net-listu preneti u odgovarajuci *qCAD* fajl sa ekstenzijom PCB čija je mesto u poddirektorijumu sa rezervisanim nazivom BOARDS. Sada možemo kao i ukoliko nemje shema samo skicirana na papiru (najčešci slucaj) odmah da predemo na program za editovanje ploče – PCBEDIT.EXE.

Za rad editora je neophodno da u tekucem direktorijumu imamo biblioteku PACKAGE.LIB koja treba da sadrži definiciju svih elemenata koji se ne ploci koriste. Ukoliko neki elemenat nije definisan pre editovanja ploče treba pozvati program za editovanje biblioteke – LIBEDIT.EXE.

Pošto je PACKAGE.LIB organizovan kao običan ASCII fajl i njegov editor *LIBEdit* je u



Rad sa bibliotekom elemenata



Originalno označavanje izvoda

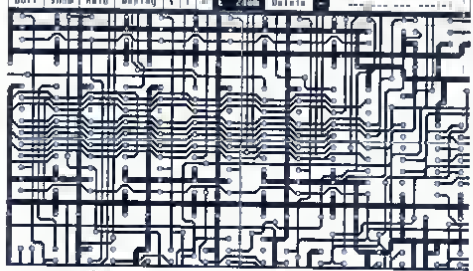
stvari standardan editor teksta (može se kao lakav i nezavisno koristiti – ima sve standardne mogućnosti i veoma je brz). Po određenom protokolu se definiše raspored numeracija, veličina i vrsta pedova (sve dimenzija se daju u milima), kao i visla alata koji će se koristiti za bušenje rupa (brojevima 1-6 za rupe ili 0 ako nema bušenja – za SMD tehnologiju na primer). Takođe se definiše i okvir elemenata (outline) da bi se znalo koliko prostora taj elemenat zauzima na ploči. Posebna opcija „Test library" *LIBEdit*-a testira ispravnost definisanje svakog elementa biblioteke (sintaksu i prolaz kol) a moguće je proveriti i konačan grafički prikaz svakog elementa – preko opcije „Show package" (slika 1)

PROJEKTOVANJE PLOČE

Program *PCBEdit* sadrži sve što je potrebno za precizno definisanje budućeg štampane ploče određuje se veličina ploče broj slojeva, oblik i orijentacija elemenata na ploči mesto vrsta i veličina zabranjenih zona na ploči (zone u kojima autoruter neće poviaciti veze) definiša se koji će signali (obično napojni) biti izvučeni debljim vezama da li će autoruter raditi u standardnom režimu (veze na rasteru 1/20 inča) ili sa vecom gustinom veza (*high density routing*)

Priprema štampane ploče počinje definisanjem naziva i veličine ploče kao i izborom broja slojeva jednoslojna jednosloje sa dzampernima (posebno zanimljiva opcija za prototipske „amaterski" radene ploče) dvoslojna dvoslojna sa posebnim GND i VCC slojem (ukupno 4 sloja) četvoroslojne i četvoroslojne sa posebnim GND i VCC slojem (ukupno 6 slojeva) Ove mogucnosti uz izvanredno efikasan i brz autoruter mogu da zadovolje prakticno sve profesionalne zahteve

Nastavak posla se sastoji u pozivanju odgovarajućih elemenala iz biblioteke (opcija Create). bilo da se radi o redefinisanju već unetih elemenata preko programa *Import* ili o novom definisanju direktno sa sheme Ukoliko ne znamo napamet kakav naziv ima elementi u biblioteci možemo koristiti opciju **List Library Packages** i izabrati željeno kuciste (PAJR1 2 3 za dvopinske elemente – otporni ke kondenzatore DJL4 8 10 za standerdne integrisana kola itd) Element se pri kreiranju postavlja na rasteru 1/10 inča sa orijentacijom 0 90 180 ili 270 stepeni e ako je potrebno moguće ga je naknadno pomeriti u bilo kom smeru sa rezolucijom do 1 mila U radu sa elementima moguće je koristiti sve ucbicajene opcije (pomeranje rotacija kopiranje brisanje kako pojedinačnih elemenata tako i blokova) a posebnosti i prednost je u tome što opcije pomeranje i kopiranja ostaju aktivne u svim menijima i ostvaruju se samo polezom miša.



Ručno crtanje veza

Za svaki element je moguće definisati naziv (**Element** U1 U2 C12 R35) tip kucišta – oznaka iz biblioteke PACKAGE LIB (**Package** DIL14 PLCC52 PAJR2 PAJR4) i komentari (**Comment** 74LS00 68HC11 1uF/16V 10 KOhm)

Prednost ovog editora nad konkurentima je u tome što ne zahteva prethodnu zamelnu pripremu nel-lista (obično preko nekog teksl editora) već se ona dobija automalski (preko programa *Import* iz odgovarajuceg *pCAD* ili *OR CAD* fajla) ili se unosi interaktivno u okviru **Modify** opcije *PCBEdit*-a Naime svaki pin odgovarajuceg elemenla je prikazan kao kvadrel u koji se interaktivno unosi ime signala (do 6 alfanumerika) Ovim se ostvaruje izuzetna lakoća i preglednost editovanja slampane ploče koju nemamo kod sličnih programa (slika 2) U okviru ovoga je i jednostavno zumiranje i izbor razmere (10 mogucnosti) korišćenjem samo jednog tastera na tastaturi Tu je i uobičajena opcija **Find signal** gde se simulirano prikazuju svi pinovi sa istim signalom i u uglu njihov ukupan broj na ploci Kriterijum pri opciji **Find** po red signala može biti i naziv elementa pako vanje ili komentar

Ako želimo da pojedini signali budu izvuceni debljim linijama (prvanstveno napojni signali zbog većih vrednosti struja kroz njih) možemo ih definisati u posebnoj tebeli pa će autoruter voditi računa i o lome

Rad sa blokovima je uglavnom standardan Poted pomeranja rotacije kopiranja i brisanja tu je i funkcija zamene stringa koja je posebno korisna posle kopiranja blokova koji se ponavljaju na ploči gde je često potrebno promeniti prefikse ili sufikse u nazivu signala Konsna blok-opcije je automatsko pozicioniranje elemenate u okviru izadranog bloka (koji moze predstavljati i celu ploču) gde se na osnovu kriterijuma minimalne ukupne dužine veza vrši automatsko raspoređivanje elemenala na ploči lako ova opcija nije svemoguca može prilično da olakše posao autoruteru posebno kod manje iskusn h korisnika

Za sve vreme rada u editoru u uglu ekrane možemo pratiti podatak o ukupnoj dužini veze i tej podatak je veoma korisan kod odlucivenja o konacnom pozicioniranju elemenala

Ukoliko na slampanoj ploči treba predvideti određene zone u kojima ne sme da bude veze (mesta za prolaz zavrtnja za konektore hlad njake ili unošenje odredenih oznaka i tekslova kao i zone koje se isecaju kod ploca nepravilnog oblika) koristimo opcije za definisanje zabranjenih zone (*obstacles*) Broj ovih zona nije ogranicen i mogu se definisati 3 lipa one koje obuhvalaju sve slojeve (A) one koje su samo na strani elemenels (E) ili samo na sireni za lemljenje (S)

Na kraju rada sa editorom može se koristiti opcija **Test Board** pri čemu program prijavljuje sve eventualne greške (na primer ako se signal pojavljuje samo na jednom pinu – nema odradista prekidapju se element i zabranjena zona element je van ploče zabranjena zona van ploce) Pri ovome se automatski formira i poseban ASCII faj LAYOUT ERR koji sadrzi sve potrebne podatke o greskama

Kada je konacno sve završeno sve greske otklonjene korisnik zadovoljan onim što je napravio ostaje samo da se mišem „klikne" na fajl opciju **Prepare data for routing** u kojoj aditor formira pel lajlova sa imenom LAYOUT i odgovarajucom ekstenzijom u podditektorijumu sa rezervisanim imenom STATES a na osnovu kojih će autorutel obavi li svoj deo posla

RUČNO DOTERIVANJE

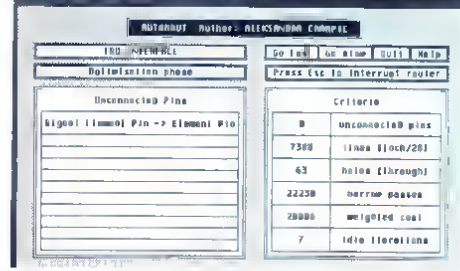
HandRout se koristi samo u slučaju kada iz nekih posebnih razloga ze mo da unapred povučemo neke veze koje autoruter kasnije neće „dirati" Naravno ko zeli može i celu ploču povezati ručno i pri tome imati na raspolaganju sve potrebne opcije (tanke i debije linije proizvoljne pedove horizontalno vertikalno i kose) povlačenje linija zumiranje slike prikaz jednog ili više slojeva simultano) Tu je i posebno korisna opcija **Record – Replay** kojom se može zapamtiti sekvenca kolisccnja tastera za pomeranje kursora prilikom isctlavanja odredene veze pa želim samo jednim klikom" sve ponoviti samo gde je potrebno (dealno za brzo isctlevanje „Bus" veza kod digitalnih uredaja) (Kako sve to izgleda moze se videti u prikazu ne slici 3)

Naravno sev rad korisnika je pod punom kontrolom programa **Test** opcija će proveriti sve veze na osnovu podataka iz *PCBEdit* a i prijaviti svaku eventualnu grešku koju korisnik prilikom povezivanja napravi

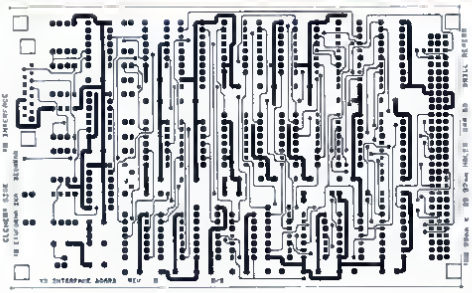
HandRout se može koristiti i posle autorutera ali iz dugog iskustva sa ovim programskim paketom tvrdimo da posle rada autorutera uz dovoljno vreme optimizacije necete moći da smislite ni jedno bolje rešenje veza od onog koje je dao sam program!

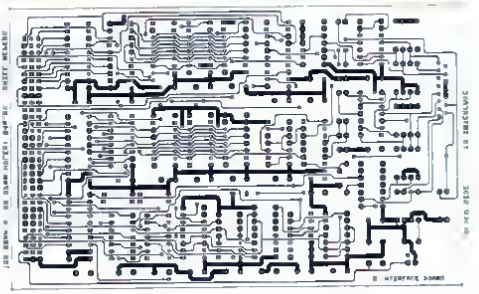
BEZ UJEDNE GREŠKE

Ako neko od korisnika koji upotrebljavaju *pCAD TANGO* ili neki slican svetski poznat programski paket za projektovanje štampanih ploča bude imao po neku primedbu na pojedine programe iz paketa *qCAD* lo se siguino neće odnositi na *AutoRout* Po brzini i pre svega po efikasnosti rešavanje veze optimizacija *AutoRout* iz *qCAD* paketa nema premca! Posle trogodisnjeg rada sa *qCAD* autorulerom



Spisak nepovezanih pinova u prvoj iteraciji





Konačni izgled štampane ploče (umanjeno)

obavljenih poređenja odgovorno tvrdimo da značajan broj ipak složenijih ploča koje *qCAD* bez problema u potpunosti (100%) i elikasno resava *pCAD* ili posebno *TANGO* neće rešiti ni sa 70% povezanih signala. Nadamo se da ovaj podatak sve govori!

Posle završenog editovanja ploče i generisanja LAYOUT * .lajlova u poddirektorijumu STATES konsnuku ostaje samo da pozove AUTOROUT.EXE i startuje program. Ako se radi o jednostavnijoj dvostranoj ploči „Evropa" formata (160 x 100 mm) sa „samo" 15-20 integrisanih kola i nekoliko desetina diskretnih elemenata (otpornici i kondenzatori) prvo rešenje ćemo imati za samo nekoliko sekundi! Naravno, moguće je da u toj prvoj iteraciji sve linije i ne budu povezane (spisak nepovezanih pinova imamo na ekranu – slika 4). Međutim, program nastavlja da radi i ukoliko još sve nije povezano ulazimo u **Retrace** fazu u kojoj se promenom rasporeda povezivanja i paralelnom optimizacijom dolazi do punog rešenja – 100% povezivanja. Ova faza može da traje od nekoliko sekundi do nekoliko desetina minuta – sve zavisi od složenosti ploče.

Kada se dođe do punog rešenja veza (100% povezivanja) program ulazi u fazu optimizacije koja traje beskonačno – stvar je korisnika kada će prekinuti izvršenje programa i upisati rešenja na disk. Kriterijum za to je broj „jalovih" iteracija (podatak imamo na ekranu) ako posle nekoliko stotina iteracija program ne dolazi do boljeg rešenja, može se proceniti da je dalja optimizacija nepotrebna.

Kriterijum za optimizaciju je ukupna dužina veza, broj prolaznih rupa i broj kritičnih prolaza (mali razmak između peda i linije ili linije i linije). Svaki od ovih elemenata učestvuje u izračunavanju ukupne dužine rešenja i svako rešenje sa manjom dužinom predstavlja poboljšanje ploče.

Kako je *AutoRout* obavio posao možete odmah proveriti ulaskom u *HandRout* i eventualno doterati pojedine veze (što će retko biti potrebno). Konačan izgled svih slojeva, belog sloja zelenih maski i dril-mape možete proveriti formiranjem grafičkih meta fajlova.

ZAVRŠNI RADOVI

Program *MetaMake* služi za generisanje grafičkih meta fajlova (sa ekstenzijom MET) u poddirektorijumu sa rezervisanim imenom METTA a na osnovu podataka iz LAYOUT * .lajlova iz poddirektorijuma STATES. Po pozivanju programa METAMAKE.EXE sve što treba da uradimo je da odredimo koje * MET fajlove želimo da generišemo i sa kojim parametrima.

Program može da generiše sledeće grafičke fajlove: signalnu mapu (identičnu onoj koju vidimo u *PCBEdit* programu), belo sito (sadrži raspored i oznake svih elemenata na ploči), dril-mapu (daje repere za bušenje rupa), dril legendu (sa različitim oznakama za razne alate definisane uz elemente u PACKAGE.LIB) zelenu masku na strani elemenata, signalni sloj na strani elemenata, unutrašnje signalne slojeva (do celih sloja), signalni sloj na strani za lemljenje i zelenu masku na strani za lemljenje. Uz sve ovo može da se generiše i posebam ASCII fajl „NC Drill Exelon" koji sadrži popis alata i odgovarajuće koordinate svih rupa, i može se koristiti za automatsko upravljanje numeričkom bušilicom pri izradi ploče.

Parametri koji se dalinišu su širina tankih linija, širina debljih (napojnih) linija, prečnik prolaznih pedova, uvećanja prečnika pedova elemenala (u odnosu na zadato u PACKAGE.LIB), uvećanje otvora na zelenoj maski (u odnosu na odgovarajući ped) i uvećanje prečnika pedova u napojnim slojevima. Sve dimenzije se zadaju u milima (korak je 1 mil). Postoji i posebna opcija **Auto** kojom se zadaje da se prilikom generisanja * MET fajlova pedovi automatski povećavaju ako u njihovoj okolini za to ima mesta. Takođe se može birati da li će kod formiranja zelene maske prolazni pedovi biti zaštićeni ili ne.

Posle generisanja * MET fajlova možemo koristiti *MetaEdit* da izvršimo pojedine intervencija na njima. Ovo se pre svega odnosi na unošenje nekih oznaka i tekstova (ime proizvođača, broj štampana ploče, imena signala pored pedova ili provodnika ...).

Na raspolaganju nam je širok izbor mogućnosti delinisanje sirine linija, prečnika pedova, zamene postojećih linija i pedova lanjim ili debljim, izbor veličine debljine i orijentacije slova, kopiranje, translacija, rotacija ploče i pojedinih blokova u ploči, crtanje krugova i kružnih lukova, kosih linija pod bilo kojim uglom, zumiranje.

Kao što se vidi, imamo jedan uglavnom standardan program za crtanje u vektorskoj grafici koji se i nezavisno može koristiti.

Posebnu snagu *qCAD* paketu daje mogućnost i isključiranja svih intervencija koje smo obavili u *MetaEdit*-u. Ovo će uraditi *MetaTast* program. Ako smo prilikom editovanja prekinuli neku vezu ili spojili dva ili više signala, program će nas na to upozoriti. Takođe se ispituje bliskost veza i pedova, ako je manja od one koju smo zadali na početku *MetaTast*-a, program će izvršiti *automatsko* razmicanje! Kakav ovo komfor i sigurnost u radu pruža – nije potrebno posebno naglašavati.

Konačan izgled pojedinih slojeva štampane ploče dat je na slici 5. Vidi se da pored svih tehničkih uslova krajnji rezultat rada *qCAD*-a zadovoljava čak i sve estetske kriterijume. Ovo može biti i subjektivna ocena, ali je činjenica da se na osnovu izgleda ploče lako može zaključiti da je nastala radom mašine a ne radom nekog iskusnog projektanta.

Jove želimo da ganensemo i sa kojim parametrima.

STIGLI SMO DO CILJA

Kada smo sa *MetaEdit*-om uradili sve što smo želali na konačnom dotenvanju izgleda štampane ploče, ostaje nam samo da sve to prenesemo na papir, konsteci program *Meta Draw*. Od kvaliteta štampača ili plotera koji ćemo koristiti kao izlazni uređaj zavisi da li će ceo nas trud i mogućnosti *qCAD*-a biti adekvatno preneseni na filmove za izradu štampane ploče. Zato je potrebno imati kvalitetan štampač ili ploter ili koristiti veće razmere štampanja kod slabijih štampača (9 pinskih na primer).

Program omogućava rad sa širokom lepezom izlaznih uređaja: od 9-pinskih matričnih štampača do specijalizovanih foto-plotera. Mogućnosti su mu uglavnom standardne – definisanje veličine papira, precizne razmere štampanja, štampanje crteža iz više delova, rotacija i štampanje crteža u ogledalu, redirekcija izlaza na serijski ili paralelni port, disk ili ekran. Moguće je uneti i prečnik iglice kod matričnih štampača ili prečnik pera kod plotera. Iako da program može automatski da unosi korekciju kako bi se dobila precizna debljina linija ili prečnik peda, onako kako je to definisano prilikom generisanja grafičkih fajlova sa *MetaMaka* i *MetaEdit* programima.

Na kraju želimo ponovo da naglasimo da *qCAD* predstavlja veoma kvalitetan softverski proizvod sa visokim performansama. Do sada je zahvaljujući *qCAD* paketu izrađeno nekoliko daseina veoma složenih i u svakom pogledu kvalitetnih štampanih ploča za profesionalna uređaja. Posebno ističemo brz i lak put od prototipske do konačne verzije ploče, zahvaljujući lakom editovanju i brzom i elikasnom autorutiranju.

Koliko nam je poznato, rad na *qCAD* paketu se nastavlja. Iako da kod novih izdanja možemo očekivati još prijatnih iznenađenja.

Pored IBM PC kompatibilnih, *qCAD* paket je dostupan i vlasnicima ATARI ST računara.

Jako prvenstveno napisan za profesionalce u ovoj oblasti, koriano može da posluži i amaterima. Za sve informacije o programskom paketu, kao i uslovima nabavke, kako demo iako i radne verzije programa, obratite se autoru *qCAD*-a.

Korisna adresa

Aleksandar Čarapić
Lole Ribara 4, 11000 Beograd
Tel. (011) 346-432, 146-822

ŠKOLSKI RAM DISK

U prošlom broju smo se detaljno pozabavili konceptom programa za pogon perifernih uređaja – device driver-ima. Pokazali smo koliko je ovaj koncept odlično zamišljen i kako omogućuje da se na veoma jednostavan način u sistem uklopi niz najrazličitijih perifernih uređaja. Priču nastavljamo i u ovom broju, ali sa drugom vrstom drajvera i na nešto složenijem primeru.

Zoran Životić

Deleći *device driver*-a na one koji podržavaju karakter orijentisane uređaje (čist tok podataka bez dodatne logičke organizacije kao što je slučaj kod štampača, modema itd) i na one blok orijentisane kao što su diskovi u prošlom broju smo izabrali za primer *device driver* iz prve kategorije pre svega zbog jednostavnosti. Kao što ćemo ovim tekstom pokazati pisanje programa za podršku blok orijentisanih uređaja je samo neznatno komplikovanije. Razlika se pojavljuje samo zbog toga što je potrebno dodatno poznavanje struktura osnovnog zapisa kojim se identifikuje medij i formira logička organizacija podataka.

U prošlom broju smo naglasili da, što se samog skupa funkcija koje *device driver* treba da obavi liče, nema razlike između ova dva kategorije. Skup funkcija je potpuno standardan, ali su, naravno, prosleđeni parametri nešto drugačiji. Takođe, neke funkcije koje karakterorijentisani uređaji ne zahtevaju sada će se pojaviti, ali u vrlo ograničenom broju. U ovom tekstu nećemo ponavljati detalje oko strukture *device driver*-a, niti načina na koji se funkcije pozivaju – čitaoca upućujemo na prethodni tekst koji je neophodan da bi se se razumevanjem pratio ovaj nastavak.

Za izbor primera *block device driver*-a nemamo mnogo dilema, jedino što možemo da uradimo bez konkretnog uređaja je da konstruišemo program poznat pod nazivom RAM-DISK – simulaciju pravog, diska koja umesto magnetnog medija za zapis koristi računarevu RAM memoriju. Napisaćemo *device driver* koji će, nakon dodavanje linije u CONFIG.SYS datoteku

```
DEVICE=BDD.SYS
```

kreirati ram disk pod prvom sledećom slobodnom oznakom (ako imate hard disk C i D, naš disk će se pojaviti kao E:). Za kapacitet diska smo izabrali izuzetno malu količinu – 32K – pre svega zbog toga što konstantno osnovnu DOS memoriju umanjuje 640 K, pa bilo šteta za ovakvu probu trošiti dragocen prostor. Namera ovog teksta i primera je da vas upozna sa načinom a konkretna realizacija bi zahtevala još niz detalja, na koje ćemo se osvrnuti na kraju teksta.

Kod realizacija *device driver*-a prva dva pitanja koja se postavljaju je izgled, odnosno informacija koju treba postaviti u zaglavlje i za početak, minimalan skup funkcija koje *device driver* mora da zna kako da obavi.

Odgovor na prvo pitanje je lak – ukoliko uporedite strukturu **headar** iz ovog i prošlog primera, primetićete prvu razliku u drugoj liniji ova reč definiše atribute *device driver*-a i u njoj je najvažniji 15-ti bit – ako je 0, znači da je nas drajver *block device driver* tipa. Ostali bitovi za osnovnu varijantu drajvera mogu biti jednaki nuli, što nam daje jedan sasvim standardan *block device driver*, bez posebnih mogućnosti. Druga razlika je u području u kome je smešteno ime uređaja. U slučaju *block device driver*-a prvi bajt ovog polja sadrži broj jedinica koje ovaj drajver podržava, dok ostalih sedam bajtova moraju imati vrednost 0. Broj jedinica koje drajver podržava omogućava da jednim *device driver*-om pokrijemo više identičnih uređaja, ili da isti uređaj mapiramo kao više logičkih jedinica. Ukoliko u ovo polje upišemo recimo broj 2, na sistemu će se pojaviti dva nova diska sa oznakama susednih slova. Naravno odgovornost *device driver*-a je da ostalkom funkcija podrži ovakav rad.

ORGANIZACIJA DISKA

Pre odgovora na drugo pitanje moramo ipak da se pozabavimo informacijama koje smo pomenuli – internim strukturama kojima se identifikuje medij i formira logička organizacija. Osnovu rada sa svakim diskom čine struktura koja se naziva **Bios Parametar Block** (BPB). O jednoj njenoj kopiji vodi računa sam *device driver* i obično je preuzima iz BIOS-a za standardno podržane disk jedinice, a druga se nalazi na samom mediju i to u takozvanom *boot* sektoru koji je uvek nulti sektor diska. Ukoliko disk jedinica podržava više različitih formata kao što je to standardno slučaj sa disketnim jedinicama na PC računaru (1.2M/360K i 1.44M/720 K) za svaki od njih mora postojati odgovarajuci BPB. Na osnovu sadržaja koji pročita sa diskete, sistem je u stanju da prepozna koji je stvarno format zapisa na disketi i da, prema tome, upotrebi odgovarajuće parametre za rad.

Kao što se iz našeg primera vidi, ispred BPB u *boot* sektoru diska ide jedna masinska JMP naredba i OEM identifikacija medija. Smisao JMP instrukcije se iz primera ne vidi jer ona u našem slucaju nikada neće biti izvedena. Neće biti izvedena nikada zbog toga što se operativni sistem mora učitati sa nekog drugog medija pre nego što uopšte prihvati naš novi disk. Da je ovakav *boot* sektor stvarno na disketi, on bi bio udaljen od strane BIOS-a i polom izvršan skok na ovu JMP instrukciju. Nas je zadatak da njome preusmerimo tok na program koji treba da učita operativni sistem ili bar da prijavi da na disketi nema operativnog sistema. Zato se obično skače negde iza BPB područja, gde sa lakav program upisuje. Program naravno mora biti vrlo ograničene dužine (do kraja *boot* sektora koji je ceo 512 bajtova) ali je ovo dovoljno da obavi izražen zadatak (možete pogledati sadrzaj nultog sektora bilo koje disketa formatirane parametrom /s) a ima čak i dovoljno mesta za uglavljivanje virusa. Iako je priča o *boot* sektoru – pa i podatak da na tvrdim diskovima ovo ipak nije prva informacija na disku i da ispred toga ide particiona tabela – interesantna tema ipak će se zaustaviti ovde jer bi nas to skrenulo sa osnovne teme.

Sledeca sistemska informacija na disku je takozvana FAT tabela (*File Allocation Table*). Radi se nizu 12-to ili 16-to bitnih vrednosti (16-to bitni FAT je uveden od DOS-a 3.xx jer se pojavila potreba za pokrivanjem diskova sa više od 4096 klastera) od kojih je svaki vezan za jedna klaster diska. Klaster je inače niz grupisanih sektora koji čini minimalnu jedinicu



Listing 1

```
;
;  MS DOS BLOCK DEVICE DRIVER - DEMO
;  Goran Sivalic - Racunari
;
;  COMPILE:  masm bdd;
;            link bdd;
;            exe2bin bdd bdd sys
; INSTALL IN CONFIG sys: device=bdd sys
;
BYTESPERSECTOR    equ 512
SECTORSPERCLUSTER equ   2
ROOTDIRENTRIES    equ  16
RESERVEDSECTORS   equ   1
NUMBEROFFATS      equ   1
TOTALSECTORS      equ  80
MEDIADESC         equ 0f8h
SECTORSPERFAT     equ   1

request struc
        len     db      ?
        unit    db      ?
        cmd     db      ?
        status  dw      ?
        res     db      8 dup (?)
        media   db      ?
        addr    dd      ?
        count   dw      ?
        sect    dw      ?
request ends


code segment public  CODE

dev_drv proc    far
        assume  cs:code ds:code ss:code
        org     0

header  dd      -1
        dw      0
        dw      strategy
        dw      interrupt
        db      1,0,0,0,0,0,0,0


boot    equ     $                       ; boot record
        jmp     boot
        nop
        db      'BD    5.0'
bpb     dw      BYTESPERSECTOR
        db      SECTORSPERCLUSTER
        dw      RESERVEDSECTORS
        db      NUMBEROFFATS
        dw      ROOTDIRENTRIES
        dw      TOTALSECTORS
        db      MEDIADESC
        dw      SECTORSPERFAT

bootsiz equ      $-boot

bpbptr  dw       bpb

req_ptr dd        ?

jmp_tab dw        initialize
        dw        media_check
        dw        build_bpb
        dw        ioctl_read
        dw        read
        dw        read read
        dw        input_status
        dw        input_flush
        dw        write
        dw        write verify
        dw        output_status
        dw        output flush
        dw        ioctl write
        dw        device_open
        dw        device close
        dw        removable
        dw        output_busy


strategy proc far
        mov       word ptr cs:[req_ptr] bx
        mov       word ptr cs:[req_ptr+2],es
        ret
strategy endp


interrupt proc far
        push    ax
        push    bx
        push    cx
        push    dx
        push    si
        push    di
        push    bp
        push    ds
        push    es

        push    cs
        pop     ds

        les     di req_ptr
        mov     bl es:[di cmd]
        cmp     bl,10
        jbe     cmdok
        mov     ax 8003h
        jmp     invalidcmd
cmdok:
        xor     bh bh
        shl     bx 1
        add     bx offset jmp_tab
        call    [bx]
invalidcmd:
        les     di req_ptr
        or      ax,100h
        mov     es:[di status], ax

        pop    es
        pop    ds
        pop    bp
        pop    di
        pop    si
        pop    dx
        pop    cx
        pop    bx
        pop    ax
        ret
interrupt endp


media_check proc near
        mov    byte ptr es:[di+14] 1
        xor    ax ax
        ret
media_check endp


build_bpb proc near
        mov    word ptr es:[di+20] cs
        mov    word ptr es:[di+18],offset bpb
        xor    ax,ax
        ret
build_bpb endp


ioctl_read proc near
        mov    ax 8003h
        ret
ioctl_read endp

read proc near
        mov    ax word ptr es:[di sect]
        mov    bx word ptr bpb
        mul    bx
        push   ds
        push   ax

        mov    ax,word ptr es:[di addr+2]
        push   ax
        mov    ax word ptr es:[di addr]
        push   ax

        mov    ax,word ptr es:[di count]
        mul    bx
        push   ax

        call   movedata

        xor    ax ax


        ret
read endp

read read proc near
        mov    ax 8003h
        ret
read_read endp

input_status proc near
        mov    ax 8003h
        ret
input_status endp

input_flush proc near
        mov    ax,8003h
        ret
input_flush endp

write proc near
        mov    ax word ptr es:[di addr+2]
        push   ax
        mov    ax,word ptr es:[di addr]
        push   ax

        mov    ax word ptr es:[di sect]
        mov    bx word ptr bpb
        mul    bx
        push   ds
        push   ax

        mov    ax word ptr es:[di count]
        mul    bx
        push   ax

        call   movedata

        xor    ax,ax
        ret
write endp

write_verify proc near
        jmp    write
write_verify endp

output status proc near
        mov    ax 8003h
        ret
output status endp

output flush proc near
        mov    ax 8003h
        ret
output_flush endp

ioctl write proc near
        mov    ax,8003h
        ret
ioctl_write endp

device_open proc near
        mov    ax 8003h
        ret
device_open endp

device_close proc near
        mov    ax 8003h
        ret
device_close endp

removable proc near
        mov    ax 8003h
        ret
removable endp

output_busy proc near
        mov    ax 8003h
        ret
output_busy endp


initialize proc near
        mov   bx offset message
        add   bx 8000h
        mov   word ptr es:[di addr] bx
        mov   word ptr es:[di addr+2] cs
        mov   byte ptr es:[di media] 1
        mov   word ptr es:[di+20] cs
        mov   word ptr es:[di+18] offset bpbptr

        mov   dx offset message
        mov   ah 9
        int   21h

        push  es
        push  cs
        pop   es
        cld
        mov   di offset message
        push  di
        mov   cx,8000h
        mov   al,0h
        rep   stosb

        pop   di
        push  di
        mov   si,offset boot
        mov   cx bootsiz
        rep   movsb

        pop   di
        mov   ax [bpb+3]
        mov   bx [bpb]
        mul   bx
        add   di bx

        mov   al,byte ptr [bpb+10]
        mov   byte ptr es:[di] al
        mov   byte ptr es:[di+1] 0ffh
        mov   byte ptr es:[di+2] 0ffh

        pop   es
        xor   ax ax
        ret
initialize endp

movedata proc  near
        push    bp
        mov     bp sp
        push    si
        push    di
        push    ds
        push    es
        cld
        mov     cx word ptr [bp+8]
        les     di dword ptr [bp+4]
        lds     si dword ptr [bp+10]
        rep movsb
        pop     es
        pop     ds
        pop     di
        pop     si
        pop     bp
        ret     10
movedata endp

        even

message db  SAMPLE MSDOS BLOCK DEVICE DRIVER  13 10
        db  32K RAM Disk installed  13 10 $
dev_drv endp
code    ends
        end dev drv
```

Primer školskog *block device driver*-a

alokacije – datoteka duzine jedan bajt uvek zauzima jedan ceo klaster. Na disketama je to dva sektora ili 1024 K, dok se na tvrdim diskovima vrsi krupnije grupisanje – jedan klaster obicno cine celiri ili cak osam sektora.

Nakon FAT-a sledi podrucje osnovnog (\) direktorija u kome su zapisani podaci o datotekama na disku. Osnovni direktorij je ujedno i prva datoteka. Uz svaku datoteku u direktoriju pise na kom klasteru pocinje – nakon toga se vrednostima iz FAT tabele prati dalji rast datoteke – prvi FAT broj nosi broj sledeceg klastera koji datoteka zauzima, ovaj opet sledeceg i tako do konacne oznake za poslednji klaster.

Može vam se ucinili da potreba za poznavanjem ovih informacija odskace od našeg raznjeg insistiranja da *device driver*-a ne zna ama baš ništa o logickoj organizaciji podataka i da je njegov zadatak u osnovi samo da prenase niz podataka sa/na medij. Sasvim ste u pravu – za pisanje samog *device driver*-a ovakve informacije nam ne bi bile uopšte potrebne, ali ih ipak moramo znati jer ce zadatak rutina za inicijalizaciju biti i da zapise ova sistemske informacije kako bi sam *device driver* dobio ispravno formiran medij. S obzirom da se ispravno formiranje medija obavlja naredbom FORMAT, mogli smo izabrati i alternativni metod – ostaviti memorijsko podrucje koje smo odvojili za simulaciju diska potpuno neinicijalizovano, ali bismo zalo u drajver morali ugraditi podršku niza IOCTL funkcija, što je ipak nešto veci posao. Ove funkcije bi omogucile da naš ram disk sasvim standardno formatiramo komandom FORMAT nakon startovanja racunara, ali je naravno daleko elegantnije resenja da odmah nakon startovanja vec imamo formatiranu „disketu". Formatiranja ram diska je u svakom slucaju nepotrebna operacija, pa nam to mogucava da potpuno izbegnemo IOCTL funkcije, što ce znatno skratiti kod (o IOCTL funkcijama je bilo reci u jednom od prošlih brojeva *Racunara*).

INICIJALIZACIJA DRAJVERA

Sa poznavanjem osnovne logicke organizacije diskova mozemo krenuti u pisanje rutina za obavljanje funkcije broj 0 – Inicijalizacije *device driver*-a. U slucaju karakter *device driver*-a iz prošlog broja ova rutina je bila vise nego jednostavna – samo smo ispisali poruku o instalaciji i postavili adresu poslednjeg bajta koji drajver koristi kako bi DOS mogao da ovo podrucje markira kao rezervisano. Sada je potrebno da na izlazu postavimo još jednu dodatnu informaciju, a to je adresa niza na kojoj se nalaze pokazivaci na podrzane BPB. U našem primeru smo formirali samo jedan disk sa jednim formatom (iskoristili smo pri tome mogucnost da u memoriji formiramo sliku *boot* sektora a da iz nje iste strukture koristimo i sam BPB), pa je zato formiran niz od samo jedne adrese (**bpbptr**) i adresa log. niza zajedno sa brojem podrzanih bpb ova (u našem slucaju 1) poslavljena na odgovorajuce mesto u strukturi zahteva (vidi uvodni deo procedure **initalize**). Na poslednju adresu koju zauzima sam drajver dodali smo 32 K koliko nam je potrebno za disk i ovu adrasu postavili kao poslednju adresu koju cemo koristiti. Treba odmah reci da bi formiranje ram diska u osnovnom RAM u koji bi zauzimao više memorije zahtevalo nesto komplikovaniju tehniku alokacije prostora, jer *device driver* dozvoljava da se alocira ukupno 64 K memorije. Ipak ovu diskusiju necemo iznositi jer se stvarni ram disk retko pravi tako da koristi osnovnu memoriju – daleko pogodniji je EMS ili XMS standard na šta cemo se osvrnuti na kraju.

Nakon što smo postavili potrebna izlazne parametre i ispisali poruku o instalaciji obavicemo „ručno" posao formatiranja naše „diskete". Celo memorijsko podrucje cemo inicijalizovati nulama (sto inace nije potrebno dovoljno je da to ucinimo samo do uključujuci i njega prostora koji zauzima *root* direktorij). Zatim cemo na prostor koji predstavlja nulti *boot* sektor prepisati sadržaj koji smo staticki formirali (struktura *boot*). FAT tabela pocinje na prvom sektoru iza rezervisanog podrucja i inicijalizuje se nulama, osim prva celin bajta. Drugi i treci imaju rezervisanu vrednost 0ffh, dok je prvi lokozvani **madia descriptor** kojim se opisuje o kom standardnom (ili nestandardnom) formatu radi. Ovaj podatak prepisujemo iz samog BPB a a izabrali smo vrednost F8h, što praktino znaci da simulriramo tvrdi disk. Cevrti bajt je 0ffh ukoliko je fat 16 bitni, inace ostaje nula.

Govoreci o lokaciji FAT tabele pomenuli smo i neko „razarvisano" podrucje. Podatak o broju sektora koji su rezervisani za sistemske informacije se nalazi u samom BPB (polje RESERVEDSECTORS) i predstavlja samo otvorenu mogucnost da rezervišemo više od jednog saktora za naše potrebe – što sa osnovog standarda liče, samo jedan sektor je razervisan i to za *boot*.

Osnovni direktorij ne moramo posabno inicijalizovati jer smo to vec ucinili ranije upisujuci nule u celo memorijsko podrucje. Ovim je ujedno inicijalizacija golova – možemo postavili izlazni status na nulu (uspešno inicijalizovan drajver) i završili ovu funkciju.

Završivsi ovu funkciju zapravo smo polpuno obavili osnovni posao – ako drajvar u ovoj formi instalramo, pojaviće se oznaka nove disk jedinice u sistemu. Ipak potrebne su nam samo jos celiri funkcije pa da novi disk postane potpuno operativan. Dve ocigledne su citanje/pisanje dok su dve druge specificne za *block device driver*. Prva, pod brojem 1 je **media check** – radi se o funkciji koja treba da proveri da li je sam medij fizicki menjan od poslednjeg poziva (praktino da li je disketa vadana iz disk jedinice). Status koji mi vracamo je uvek 1, jer on oznacava da medij nije diran i da nama potrebe da se sistem dalje raspituje o tome šta se sada nalazi u disk jedinici. Drajver za pravu disk jedinicu bi ovde morao da konsultuje hardversku liniju sa disk jedinice „diskette change line", koja mu je jedini indikator da se nesto ovako desilo. Zahvaljujuci jedinstvenom serijskom broju koji se na disketu upisuje u novim verzijama DOS a (pošto se pokazalo da retko koji korisnik uopšte upisuje labelu diskete a kamoli jedinstvenu za svaki disk što je inace mogucnost da se proveri da je disketa promenjena) mogu?e je ustanoviti promenu diskete i na disk jedinicama bez ovog senzora. DOS u se može vratiti i status „nisam siguran" (vradnost 0) kada ce, ukoliko u baleru jos ima podalaka koji nisu zapisani, ipak doneti odluku da ih zapiše. Izlikujuci da u slucaju da je disketa stvarno izmenjena praktino uništi njen zadrzaj.

Raspitivanje o tome koji medij se trenutno nalazi u jedinici sistem obavlja preko funkcije broj 2 – **build bpb** (formiraj **bpb** za medij). Za datak ove funkcije u nasem slucaju je veoma lak jer treba samo još jednom da vratimo pointer na staticku strukturu **bpb**. Da se radi o izmenjivim diskelama, drajver bi ovde morao da procita **boot** sektor diskete i da vrati pokazivac na stvaran **bpb** koji je zapisan na disketi.

FUNKCIJE READ I WRITE

Konacno stižemo i do poslede dve neophodne funkcije – **read** i **write**. Na njima ce se pokazati sva elegancija *device driver* koncepta, jer je za njihovu realizaciju potrebno izuzetno malo koda. Kao parametre sistem nam prosleduje adresu svog balera u koji želi da primi podatke, pocetni sektor i broj sektora koji želi da procita. Na nama je da ovaj zahtev prevedemo u memorijsku adresu na kojoj se kod nas laj sektor nalazi i da samo kopiramo potreban broj sektora u prijemni bafer. S obzirom da **write** funkcija obavlja identican posao u suprotnom smeru, kiteirali smo odvojenu rutinu za kopiranje kojoj preko staka prosledujemo izvornu adresu, odradisanu adresu i broj bajtova koje treba kopirati, pa je nakon toga realizacija ova dva funkcija trivijalna.

U drajver smo ugradili i funkciju broj 9 – njena namena je zapisivanje sa verifikacijom. Verifikacija nije obavezna – sistem nam samo nudi mogucnost da to ucinimo, a inace pos svemu ostalom funkcija je identicna funkciji **write**. Da li ce sistem pozvali **write** ili **write with verify** zavisi od toga da li smo postavili DOS VERIFY=ON. S obzirom da nema potrebe da proveramo da li smo podatke stvarno dobro zapisali u memoriju u primeru ovu funkciju preslikavamo u obicnu **write**.

Time bi ujedno drajver bio potpuno kompletan i spreman za upotrebu. Sve ostale funkcije vracaju status: funkcija nije podrzana" ali to u upotrebi necemo primetili dok ne pokusamo da formatiramo nas ram disk – tada ce sistem prijaviti grešku jer u našem drajveru nema IOCTL funkcija koje inace obavljaju ovaj posao.

OD SKOLSKOG DO PRAVOG

Za kraj nam ostaje da pokazemo šta bi bilo potrebno menjati u ovom drajveru ako od njega zelimo da ucinimo „pravi" ram disk. Prva promena je odluka o tipu memorije koji zelimo da koristimo. Najpogodnije je da to bude EMS i XMS memorija. U tom slucaju bi u inicijalizacionoj rutini trebalo dodati alokaciju potrebne kolicine ove memorije (o radu sa EMS memorijom je bilo reci u *Racunarima*) a morali bismi i malo da iskomplikujemo rutinu **movedata**. U njoj bi na osnovu adrese sektora trebalo izracunati na kojoj se on EMS strani nalazi, dovesti je u EMS ram i potom izvrsiti kopiranje. Ovaj posao nije posebno komplikovan ako se poznaju EMS ili XMS standard.

Inicijalizaciona rutina bi morala da pretrpi i jednu nešto složeniju izmenu. Veliinu ram diska bi trebalo formirati prema raspoloživoj memorijskoj situaciji i prema parametrima zadatim kod instalacije drajvera (u strukturi zahteva pointer na liniju iz CONFIG.SYS datoteke kojom je drajver startovan nalazi se na ofsetu 18 (olsel + segment). To znaci da bi dinamicki morali da formiramo **bpb** (polje TOTALSECTORS) le da broj sektora koje ce zauzimat. Jat izracunamo na sledeci nacin. Prvo bi na osnovu ukupnog broja sektora i našeg izbora broja saktora koji cine klaster (mora biti stepen 2) izracunali ukupan broj klastera na disku. Svaki klaster zahteva 12 bita (ili 16, ako ih ima vise od 4096), pa prama tome treba izracunati koliko nam je sektora potrebno za FAT. Na stvarnim diskovima se obicno diže dve kopije FAT labela ali je to nepotrebno u slucaju ram diska. Tek nakon ovog obracuna mozemo popuniti polje SECTORSPERFAT i ujedno saznali po celni sektor za direktorij (broj rezervisanih sektora + broj sektora FAT-a). Takode, ukoliko smo ustanovili da nam je potrebna 16-bitna FAT tabela, treba ili da izmenimo 4 bajt pocetka FAT a sa 0 na 0ffh. Sve u svemu, cak i pravi ram disk drajver nece biti mnogo komplikovaniji od naseg primera.

U SREDIŠTU SISTEMA

Seriju tekstova o programiranju za *Windows* okruženje smo, kao što je to uobičajeno, počeli osnovnim objašnjenjima i trivijalnim tehnikama. Nakon toga bavili smo se raznim aspektima korisničkog interfejsa, a sada je pravo vreme da se posvetimo onome što prirodno sledi u programerskoj lestvici – sistemskom programiranju!

Dejan Dinčić

Iako *Windows* 3.0 još uvek nije ni blizu tako rasvetljen i dokumentovan kao stari poznanik DOS i često vam se čini da lutate u mraku dok programirate za *Windows* ipak postoje neki putokazi ka središtu sistema



FILTERSKE FUNKCIJE

Jedna od mogućnosti da se na legalan način kontroliše ponašanje *Windows* okruženja je upotreba **Windows Hook** (*hook* – udica kuka) tehnike. Postavljanjem **Hook**-a se instalira filter funkcija koju *Windows* poziva pre nego što obradi neki događaj. Tipičan primer: možemo postaviti sopstvenu funkciju, koja će biti pozvana svaki put pre nego što sistem isporuči poruku bilo kom prozoru u sistemu. Naša funkcija može i da modifikuje poruku pre nego što vrati kontrolu i tako utiče na rad sistema. Jasno je da će se ova funkcija pozivati jako često. Ie da ovakve intervencije treba raditi vrlo pažljivo. Zajedno sa velikim mogućnostima koje pruža ova koncepcija, pojavljuje se i opasnost usporenja rada celog sistema.

Definisano je sedam vrsta događaja za koje je moguće postaviti sopstvenu filter funkciju

WH_CALLWNDPROC – Filter funkcija se poziva svaki put kada se nekom prozoru u sistemu poruka isporučuje funkcijom **SendMessage**

WH_GETMESSAGE – Filter funkcija se poziva svaki put kada neki prozor uzima poruku funkcijama **GetMessage** ili **PeekMessage**

WH_KEYBOARD – Filter funkcija se poziva svaki put kada neki prozor uzima poruku tipa WM_KEYDOWN ili WM_KEYUP funkcijama **GetMessage** ili **PeekMessage**

WH_SYSMSGFILTER – Filter funkcija se poziva svaki put kad dijalog meni poruka, ili skrol bar treba da obradi poruku bilo kog tipa

WH_MSGFILTER – Filter funkcija se poziva svaki put kad dijalog meni poruka, ili skrol bar koji pripadaju aplikaciji koja je instalirala filter obrađuju poruku bilo kog tipa

WH_JOURNALRECORD – Filter funkcija se poziva svaki put kad se obradi događaj (event) iz sistemskog lanca događaja (*system queue*)

WH_JOURNALPLAYBACK– Filter funkcija se poziva svaki put kad se zatraži događaj (*event*) iz sistemskog lanca događaja (*system queue*)

Filter funkcije se instaliraju funkcijom **SetWindowsHook**. U sistemu može biti postavljeno više filter funkcija koje obrađuju isti tip događaja. Pravilnom instalacijom one se vezuju u lanac u kome se poslednja instalirana funkcija poziva prva, kad nastane očekivani događaj. Nakon obavljanja svog posla ona poziva sistemsku funkciju **DefHookProc**, koja prenosi kontrolu sledećoj funkciji u lancu.

Mnogi „špijunski" programi koriste alatke za traženje grešaka i analizu napisani su korišćenjem ovih funkcija. Prikazećemo na primeru (kao i uvek do sada) osnovne elemente ove tehnike.

DINAMIČKO POVEZIVANJE

Naš primer program **wkey** služi redefinisanju tastatura. Konsenje filter funkcija nije baš idealan način da se napravi ozbiljan drajver za tastaturu, ali je osnovna svrha programa da ilustruje **hook** tehniku. **wkey** se može upotrebiti i da *Windows* prilagodi vašoj (YU) tastaturi. Program se sastoji iz dva dela. Prvi deo je program wkey.exe koji inicijalizuje proces i služi za komunikaciju sa korisnikom – konfigurisanje programa. Drugi deo čini dinamička biblioteka keybyu.dll koja sadrži samu filter funkciju. Ovde je nužno prisetiti se na kom principu rade *Windows* dinamičke biblioteke (*dynamic link library* DLL)

Windows popul „pravih" operativnih sistema pruža mogućnost da se elegantno pozivaju funkcije (potprogrami) koji nisu deo programa koji se izvršava. Ako ste nekad formirali klasičnu biblioteku svojih rutina (.LIB pod DOS-om) i povezivali je uz svaki svoj program, ovde je možete formirati kao dinamičku biblioteku (.DLL) i pustiti *Windows* da izvrši povezivanje onda kad programu zatreba neka od funkcija iz biblioteke. Povezivanje se vrši u vreme izvršavanja programa (*run time*). DLL koja je učitana (*loaded*) mogu koristiti više programa istovremeno. Iako to do sad nismo posebno naglašavali, skoro sve *Windows* funkcije koje koristimo u našim programima su dinamički pozivani. Kako sva ta funkcionise?

Pri pisanju aplikativnog programa potrebno je uraditi dve stvari: učitati biblioteku u kojoj se nalaze potrebne rutine (za ovo je dovoljno pozvati funkciju **LoadLibrary**) i nekako prevanit linker da ne prijavljuje nerazrešene reference za funkcije koje će biti dinamički povezane. Uostalom, to i treba da ostanu nerazrešene referemce do trenutka izvršavanja. Da bi i linker pristao na to potrebno je u DEF datoteci (parametri linkera) dodati IMPORT sekciju i u njoj pobrojati funkcije koje se pozivaju iz dinamičkih biblioteka. Ovo će umiriti linker i on će u IMPORT tabelu u zaglavlju programa (.EXE) staviti odgovarajuće informacije

Melodi ipak nije baš previše praktican kada treba navesti veći broj funkcija (ni u dosadašnjim primerima nismo navodili spisak svih *Windows* funkcija koje koristimo). Rešenje je u korišćenju tabela koje se generišu prilikom formiranja DLL-a. Njih formira pomoćni program *Implib*. Tabela je smeštana u datoteku koja ima formu klasične biblioteke (.LIB) i sadrži sve potrebne informacije koje će linker ugraditi u IMPORT tabelu našeg programa, a ne sadrži kod biblioteke koja se koristi. Na ovaj način su obezbeđene informacije potrebne za ucitavanje DLL-a, pa se umeslo eksplicitnog poziva funkcije **LoadLibrary** biblioteka automatski učitava. To objašnjava zašto nam je bilo nepoznato da koristimo dinamičko povezivanje čak i kod upotrebe najobičnijeg **MessageBox**-a. Linker je uvek povezujuci nas program sa *Windows* bibliotekama samo koristio tabele za dinamičko povezivanje. Dovoljno je pogledati

Listing 1

```
all: keybyu.dll wkey.exe wkey.exe

wkey.res: wkey.rc wkey.h
    rc -r wkey.rc

wkey.obj: wkey.c wkey.h
    cl -c -AS -Gsw -Oas -Zpe -W2 wkey.c

wkey.exe: wkey.obj wkey.def keybyu.dll
    link /NOD wkey,,, libw slibcew keybyu, wkey.def
    rc wkey.res

wkey.exe: wkey.res
    rc wkey.res

keybyu.obj: keybyu.c keybyu.h
    cl -c -Asnw -Gsw -Oas -Zpe keybyu.c

libentry.obj: libentry.asm
    masm -Mx libentry,libentry;

keybyu.dll: libentry.obj keybyu.obj keybyu.def
    link /noe keybyu libentry, keybyu.dll , libw sdllcew keybyu.def
    rc keybyu.dll
    implib keybyu.lib keybyu.def
```

Listing *makefile*-a za celu aplikaciju

Listing 2 WKEY.C

```
/*-----------------------------------------------------------------------*/
/*  obradjuje sve poruke koje stizu wkey glavnoj proceduri              */
/*-----------------------------------------------------------------------*/
long FAR PASCAL MainWndProc(hWnd, message, wParam, lParam)
HWND hWnd;
unsigned message;
WORD wParam;
LONG lParam;
{
    int                 scancode;
    RECT                Rect;
    HANDLE              hOldBuffer;
    FARPROC             lpProcAbout;
    PAINTSTRUCT         ps;
    LPDRAWITEMSTRUCT    lpDrawButton;

    switch (message) {
        case WM_COMMAND:
            switch (wParam) {
                case IDM_ABOUT:
                    lpProcAbout = MakeProcInstance(About, hInst);
                    DialogBox(hInst, "AboutBox", hWnd, lpProcAbout);
                    FreeProcInstance(lpProcAbout);
                    break;

                case IDM_KEY_STAT:
                    wsprintf (ws, " Total keystrokes %d ", GetKeystrokes ());
                    MessageBox (hWnd, ws, " KEY HOOK ", MB_OK);
                    break;

                case IDM_KEY_DEF:
                    if (HookOn)
                    {
                        Unhook ();
                        HookOn = FALSE;
                    }
                    DefineKey (hWnd);
                    break;

                case IDM_START:
                    if (!HookOn)
                    {
                        InstallHook ();
                        HookOn = TRUE;
                    }
                    break;

                case IDM_STOP:
                    if (HookOn)
                    {
                        Unhook ();
                        HookOn = FALSE;
                    }
                    break;

                case IDM_KEY_DEL:
                    break;

                 case IDM_EXIT:
                    DestroyWindow(hWnd);
                    break;
            }
            break;

        case WM_CHAR:
            if (CatchKey == FALSE)
                break;
            CatchKey = FALSE;
            scancode = HIWORD (lParam);
            scancode = LOBYTE (scancode);
            Redefine(hWnd, scancode, wParam);
            if (!HookOn)
            {
                InstallHook();
                HookOn = TRUE;
            }
            InvalidateRect(hWnd, NULL, TRUE);
            break;

        case WM_PAINT:
            BeginPaint(hWnd, (LPPAINTSTRUCT)&ps);
            WkeyPaint(hWnd, ps.hdc);
            EndPaint(hWnd, (LPPAINTSTRUCT)&ps);
            break;

        case WM_CREATE:
            ReadCfg (hWnd);
            if (!HookOn)
            {
                InstallHook ();
                HookOn = TRUE;
            }
            break;

        case WM_DESTROY:
            SaveCfg (hWnd);
            if (HookOn)
            {
                Unhook ();
                HookOn = FALSE;
            }
            PostQuitMessage(0);
            break;

        default:
            return (DefWindowProc(hWnd, message, wParam, lParam));
    }
}

/*-----------------------------------------------------------------------*/
/*  inicijalizuje definisanje tastera                                    */
/*-----------------------------------------------------------------------*/
void DefineKey (hWnd)

HWND hWnd;
{
    HDC   hDC;
    RECT  Rect;
    WORD  TW;
    DWORD TE;

    hDC = GetDC(hWnd);
    GetClientRect(hWnd, (LPRECT) &Rect);
    strcpy (ws, " Press a key to define ");
    TE = GetTextExtent (hDC, (LPSTR) ws, strlen(ws));
    TW = LOWORD (TE);
    Rectangle(hDC,
            (short) Rect.right/2-TW/2-10,
            (short) Rect.bottom/2-10,
            (short) Rect.right/2+TW/2+10,
            (short) Rect.bottom/2+10);
    TextOut( hDC,
            (short) Rect.right/2-TW/2,
            (short) Rect.bottom/2,
            (LPSTR) ws,
            (short) strlen (ws));
    ReleaseDC(hWnd, hDC);

    /*  treba uhvatiti sledecu WM_CHAR poruku  */
    CatchKey = TRUE;
}

/*-----------------------------------------------------------------------*/
/*  menjanje opis trazenog tastera, poziva dijalog za def               */
/*-----------------------------------------------------------------------*/
void Redefine (hWnd, scode, wcode)
HWND hWnd;
int  scode;
WORD wcode;
{
    int      defined;
    struct   trkey wkey;
    FARPROC  lpProcDefKey;

    /* pronaci odgovarajuci slot tabela tastera */

    wkey = FindScanCode (scode);
    if (wkey.scan == 0)
    {
        MessageBox (hWnd, " To many key definitions ", " WKEY ", MB_OK);
        return;
    }

    /*  priprema za dijalog kojim korisnik  */
    /*  unosi znacenje kode te tastera       */

    if (wkey.outn == 0)
        wkey.outn = scode;
    if (wkey.outs == 0)
        wkey.outs = scode;
    if (wkey.outc == 0)
         wkey.outc = scode;
    key_norm[0] = wkey.outn;
    key_norm[1] = 0;
    key_shift[0]= wkey.outs;
    key_shift[1]= 0;
    key_ctrl[0] = wkey.outc;
    key_ctrl[1] = 0;
```

Listing se nastavlja na sledecoj strani

Nastavak sa prethodne strane

```
    /*  pozo dijaloga  */
    lpProcDefKey = MakeProcInstance(DefKey  hInst);
    bDlined = DialogBox(hInst  "DefKeyBox"  hWnd, lpProcDefKey);
    FreeProcInstance(lpProcDefKey);
    if (!bDlined)
        return;

    /*  modifikovati sadrzaj tabele  */
    wkey.outn = key_norm[0];
    wkey.outs = key_shift[0];
    wkey.outc = key_ctrl[0];
    modifyKeyCode(wkey);
}

/*-----------------------------------------------------------------------*/
/*   Obradjuje poruke za DefKey dialog box                               */
/*-----------------------------------------------------------------------*/
BOOL FAR PASCAL DefKey(hDlg  message, wParam  lParam)
HWND hDlg;
unsigned message;
WORD wParam;
LONG lParam;
{
    switch (message) {
        case WM_INITDIALOG:
            SetDlgItemText(hDlg  IDC_DEFKEY_NORM  key_norm);
            SetDlgItemText(hDlg  IDC_DEFKEY_SHIFT key_shift);
            SetDlgItemText(hDlg  IDC_DEFKEY_CTRL  key_ctrl);
            SetFocus(GetDlgItem(hDlg  IDC_DEFKEY_NORM));
            return (FALSE);

        case WM_COMMAND:
            switch (wParam) {
                case IDOK:
                    GetDlgItemText(hDlg  IDC_DEFKEY_NORM, key_norm  5);
                    GetDlgItemText(hDlg  IDC_DEFKEY_SHIFT,key_shift,5);
                    GetDlgItemText(hDlg  IDC_DEFKEY_CTRL  key_ctrl  5);
                    EndDialog(hDlg  TRUE);
                    return (TRUE);

                case IDCANCEL:
                    EndDialog(hDlg  FALSE);
            }
        break;      return (FALSE);
  }
  return (FALSE);

/*-----------------------------------------------------------------------*/
/*   Ucitava sig iz WKEY INI                                             */
/*-----------------------------------------------------------------------*/
int ReadOrig (hWnd)
HWND  hWnd;
{
    int  i;
    int  hFile;
    HCURSOR  hSaveCursor;
    OFSTRUCT OfStruct;                          /* informacije iz OpenFile() */
/*-----------------------------------------------------------------------*/
/*   Ucitava sig iz WKEY INI                                             */
/*-----------------------------------------------------------------------*/
int ReadCfg (hWnd)
HWND  hWnd;
   int  i;
   int  hFile;
   HCURSOR  hSaveCursor;
   OFSTRUCT OfStruct;                     /* informacije iz OpenFile() */
   struct trkey  wkey;

   if ((hFile = OpenFile(CfgFile, (LPOFSTRUCT) &OfStruct
          OF_READ)) == -1) {
      wsprintf(ws, "Error %d opening %s "  OfStruct.nErrCode,
(LPSTR)CfgFile);
      MessageBox(hWnd  ws, NULL  MB_OK | MB_ICONHAND);
      return (FALSE);
   }
   hSaveCursor = SetCursor(hHourGlass);
   for (i=0; i<MAX_KEYS && read(hFile  &wkey  sizeof(wkey))==sizeof(wkey)
i++)
      ModifyKeyCode (wkey);
   close(hFile);
   SetCursor(hSaveCursor);                         /* vrati orig  cursor */

   InvalidateRect(hWnd, NULL, TRUE);               /* precrtaj ponovo    */
   return (TRUE);
/*-----------------------------------------------------------------------*/
/*  Snima nov sig u WKEY INI
/*-----------------------------------------------------------------------*/
int SaveCfg(hWnd)
HWND hWnd;
{
    int  i;
    int  hFile;
    OFSTRUCT OfStruct;                    /* informacije iz OpenFile() */
    struct trkey  wkey;

    if ((hFile = OpenFile(CfgFile, (LPOFSTRUCT) &OfStruct,
            OF_CREATE)) == -1) {
       wsprintf(ws  "Error %d opening %s "  OfStruct.nErrCode,
(LPSTR)CfgFile);
       MessageBox(hWnd, ws  NULL  MB_OK | MB_ICONHAND);
       return (FALSE);
}

hSaveCursor = SetCursor(hHourGlass);
for (i=0; i<GetKeysNumber(); i++)
{
   wkey = GetKeyDef(i);
   write(hFile  &wkey, sizeof(wkey));
}
   close(hFile);
   SetCursor(hSaveCursor(;                          /* vrati orig  cursor */
   return (TRUE);
}
/*-----------------------------------------------------------------------*/
/*   Iscrtava prozor definisanim tasterima                               */
/*-----------------------------------------------------------------------*/
void wkeyPaint (hWnd  hDC)
HWND  hWnd;
HDC   hDC;
{
    int  i, kn;
    int  k_x = 50;
    int  k_y = 40;
    RECT Rect;
    HANDLE hBigFont,
           hOldFont;
    HBRUSH hGrayBrush;
    LOGFONT       lF;
    struct trkey  wkey;

    GetClientRect(hWnd  (LPRECT) &Rect);
    memset (&lF  0, sizeof(LOGFONT));                          /* izbor fonta  */
    strcpy (lF.lfFaceName  "Swiss II");
    hBigFont = CreateFontIndirect ((LPLOGFONT) &lF);
    hOldFont = SelectObject (hDC hBigFont);

    hGrayBrush  = (HBRUSH) GetStockObject(LTGRAY_BRUSH);
    SelectObject(hDC  hGrayBrush);
    SetBkMode (hDC  TRANSPARENT);
    kn = GetKeysNumber ();
    for (i=0; i<kn; i++)
    {
       /*  crtaj taster  */
       RoundRect(hDC,
                 (short) (i%10)*k_x+20,
                 (short) (i/10*k_y)+20,
                 (short) (i%10)*k_x+20+k_x-2
                 (short) (i/10*k_y)+20+k_y-2
                 12  12);
       wkey = GetKeyDef(i);
       wsprintf(ws, " (%d) ", wkey insno);
       TextOut( hDC,
                 (short) (i%10)*k_x+22,
                 (short) (i/10*k_y)+22
                 (LPSTR( ws,
                 (short) strlen(ws));
       wsprintf(ws, "%c-%c-%c"  wkey outn  wkey outs  wkey outc);
       TextOut( hDC
                 (short) (i%10)*k_x+22,
                 (short) (i/10*k_y)+35,
                 (LPSTR) ws,
                 (short) strlen(ws));
```

Listing dela izvornog koda za program *wkey*

Listing 3

```
/*************************************************************************

    PROGRAM: keybys c
             Sadrzi funkcije biblioteke(DLL) koje obradjuju hooks

*************************************************************************
*/

#include "windows h"
#include "keybys h"

static int          TransKeys = 0;

static FARPROC    NextHookFunc = NULL;
static HANDLE     hInstance = NULL;
static WORD       keystrokes = 0;
static struct trkey TrTbl(MAX KEYS);

/*-----------------------------------------------------------------------*/
/*  Inicijalizacija DLL a  Poziva se automatski po ucitavanju DLL-a    */
/*  U ovom slucaju samo inicijalizuje tabelu tastera                    */
/*-----------------------------------------------------------------------*/
int FAR PASCAL LibMain(hModule, wDataSeg  cbHeapSize  lpszCmdLine)
HANDLE  hModule;
WORD    wDataSeg;
WORD    cbHeapSize;
LPSTR   lpszCmdLine;
{
    hInstance = hModule;

    memset (TrTbl, 0, sizeof(TrTbl));
    return 1;
}

/*-----------------------------------------------------------------------*/
/*  Uklanja (cleanup) DLL-a  Poziva se automatski pri izbacivanju      */
/*  iz memorije  Trebe da postoji u DLL-u i ne radi nista              */
/*-----------------------------------------------------------------------*/
int FAR PASCAL WEP (bSystemExit)
int  bSystemExit
{
    return 1
}

/*-----------------------------------------------------------------------*/
/*  Funkcija Hook  poziva se uvek kad Windows treba da prosledi         */
/*  poruku nekoj aplikaciji                                             */
/*-----------------------------------------------------------------------*/
DWORD FAR PASCAL KeyHook (int nCode, WORD wParam  LONG lParam)
```

```
    LPMSG  ms;
    int    scancode;
    DWORD  dwRes = 0;

    ms = (LPMSG)lParam;
    if (nCode == HC_ACTION &&
        ms->message == WM_CHAR)
    {
        keystrokes++;
        scancode = HIWORD (ms->lParam);
        scancode = LOBYTE (scancode);
        if (TrTbl[scancode] inscan)
        {
            if ((GetKeyState (VK_CAPITAL) & 0x01) ||
                (GetKeyState (VK_SHIFT) & 0x80))
                ms->wParam = TrTbl[scancode] outs;
            else
            if (GetKeyState (VK_CONTROL) & 0x80)
                ms->wParam = TrTbl[scancode] outc;
            else
                ms->wParam = TrTbl[scancode] outn;
        }
    }

    /*   poziv sledeće funkcije u lancu   */
    if (nCode < 0 || NextHookProc != NULL)
        dwRes = DefHookProc (nCode, wParam, lParam, NextHookProc);

    return(dwRes);
}

/*-------------------------------------------------------------------------*/
/*  Stavlja filter funkciju u lanac, aktivira je                           */
/*-------------------------------------------------------------------------*/
int FAR PASCAL InstallHook (void)
{
    NextHookProc = SetWindowsHook (WH_GETMESSAGE, (FARPROC)KeyHook);
    return(1);
}

/*-------------------------------------------------------------------------*/
/*  Skida filter funkciju iz lanca, deaktivira je                          */
/*-------------------------------------------------------------------------*/
int FAR PASCAL UnHook (void)
{
    UnhookWindowsHook (WH_GETMESSAGE, (FARPROC)KeyHook);
    return(1);
}

/*-------------------------------------------------------------------------*/
/*  Vraca strukturu koja opisuje jedno taster, on osnovu scan koda         */
/*-------------------------------------------------------------------------*/
struct trkey FAR PASCAL FindScanCode (int scode)
{
    struct trkey wtr;

    if (scode >= MAX_KEYS-1)
    {
        wtr.inscan = 0;
        return (wtr);
    }
    if (TrTbl[scode].inscan)
        return (TrTbl[scode]);

     /* scode nije u tabeli  sedeti se */
    TrTbl[scode].inscan = scode;
    TrTbl[scode].outn = 0;
    TrTbl[scode].outs = 0;
    TrTbl[scode].outc = 0;
    TrTbl[scode].setc = 0;
    TransKeys++;
    return (TrTbl[scode]);
}
```

```
}

/*-------------------------------------------------------------------------*/
/*  Modifikuje strukturu koja opisuje jedno taster                         */
/*-------------------------------------------------------------------------*/
int  FAR PASCAL ModifyKeyCode (struct trkey wk)
{

    if (wk.inscan >= MAX_KEYS-1)
        return (FALSE);

    if (!TrTbl[wk.inscan].inscan)
        TransKeys++;
    TrTbl[wk.inscan] = wk;
    return(TRUE);
}
```

Listing dela izvornog koda za biblioteku, filter funkcija

Listing 4

```
LIBRARY     keyby

EXETYPE     WINDOWS

CODE        PRELOAD MOVEABLE DISCARDABLE
DATA        PRELOAD SINGLE

HEAPSIZE    1024

EXPORTS
    WEP                 @1 RESIDENTNAME
    KeyHook             @2
    UnHook              @3
    GetKeyStrokes       @4
    FindScanCode        @5
    ModifyKeyCode       @6
    InstallHook         @7
    SetKeybef           @8
    GetKeyBoeder        @9
```

Listing definicija povezivanja (link) za biblioteku

Listing 5

```
/*-------------------------------------------------------------------------*/
/*  Struktura koja opisuje redefinicije jednog tastera                     */
/*-------------------------------------------------------------------------*/
struct trkey {
  int inscan;
  int outn;
  int outs;
  int outc;
}

/*-------------------------------------------------------------------------*/
/*  Funkcije                                                               */
/*-------------------------------------------------------------------------*/
DWORD FAR PASCAL KeyHook (int nCode, WORD wParam, LONG lParam);
int   FAR PASCAL InstallHook (void);
int   FAR PASCAL UnHook (void);
WORD  FAR PASCAL GetKeyStrokes (void);
struct trkey FAR PASCAL FindScanCode (int scode);
int   FAR PASCAL ModifyKeyCode (struct trkey);
int   FAR PASCAL GetKeyBoeder (void);
struct trkey FAR PASCAL SetKeybef (int);
```

Listing zaglavlja biblioteke, sa opisom strukture koja definise taster

veličinu izvršnog koda koji smo dosad formirali i videli da se tu ne krije kod svih onih lepih grafičkih funkcija.

Svi tipovi filter funkcija moraju biti u okviru nekog DLL-a, jer filter funkcija ne može sebi dozvoliti da bude u neaktivnoj aplikaciji (npr. trenutno bez fokusa). Jedini izuzetak je da se filter funkcija za WH_MSGFILTER može nalaziti i u aplikaciji, jer nadgleda aktivnost same aplikacije.

REDEFINISANA TASTATURA

Koji tip događaja obrađivati? WH_KEYBOARD prvenstveno služi za nadgledanje aktivnosti tastature. Povratna vrednost funkcije može da utiče jedino na to da se pritisak na taster ne prikaže, ali ne može da menja vrednosti koda za pritisnuti taster pre nego se poruka prosledi dalje. Ostaje nam da iskoristimo najopširniju varijantu, WH_GETMESSAGE. Istina, naša filter funkcija će biti pozvana za bilo koji tip poruke koja se šalje nekom prozoru, ali obrađivaćemo samo poruke tipa WM_CHAR. Ova poruka se šalje prozorima kad god je pritisnut taster koja ima svoju ASCII interpretaciju, što je dovoljno za početak. Ostacemo ukraćeni samo za kurzorske i funkcijske tastere. U opštijem slučaju mogli bismo nadgledati i poruke tipa WM_KEYDOWN, a postaviti i obradu za drugi tip događaja – WH_CALLWNDPROC.

Sama filter funkcija (u našem primeru **KeyHook**) ima tri argumenta, koji zavise od tipa **hook**a. Prvi argument uvek predstavlja kod akcije koja se zahteva. Dve njegova vrednosti (HC_LPFNNEXT i HC_LPLPFNNEXT) su rezervisane za uklanjanje funkcije iz lanca filtera. Treća vrednost u našem slučaju, za WH_GETMESSAGE **hook**, može biti samo HC_ACTION i ona znači da je naša funkcija (**KeyHook**) pozvana upravo zato što neka poruka treba da se isporuči. Adresa poruke je u tom slučaju u trećem argumentu funkcije. Filter funkcija mora obavezno da pozove i **DefHookProc** funkciju, koja će obezbediti izvršavanje ostalih funkcija u lancu i normalan rad sistema.

Osim filter funkcije (**KeyHook**) modul key.byu dll sadrži i funkcije za inicijalizaciju (**LibMain** poziva kod učitavanja biblioteke) i komunikaciju sa aplikacijskim programom. Key.byu DLL je nezavisna celina, ali je za njen rad neophodan upravljački program (wkey.exe) koji će obezbediti tabelu konverzija tastera. Ovaj progam koristi uglavnom tehnike i funkcije koja smo u ranijim primerima prikazali. Osnovni mu je zadatak da formira niz struktura u memoriji koji će koristiti filter funckija za konverziju tastera. Konverzija je izvedena tako što svakom *scan* kodu (iz BIOS a) dodeljujemo ASCII vrednosti koje želimo da ima kad je taster pritisnut sam, sa šiftom ili Ctrl tasterom. Za utvrđivanje stanja šift tastera koristimo *Windows* funkciju **GetKeyState** jer, začudo, ova informacija nije data uz poruke tipa WM_KEYDOWN i WM_CHAR.

Interesantno je da *Windows* ume da „izvadi" funkciju iz lanca nizom poziva ulančanim funkcijama, bez obzira na položaj funkcije u lancu (početak, sredina, kraj lanca).

Ostaje još samo da vam preporučimo da programiranju ovakvih rešenja, sa toliko moći na raspolaganju, pristupite dovoljno pažljivo, tako da se ne naruše performanse ili stabilnost celog sistema. Onda možete pustiti masti na volju i pisati sistemske programe za *Windows* koliko vam srce želi.

RASTAVI ME NEŽNO

U seriji od nekoliko tekstova biće predstavljeni programi za tzv. „simboličku manipulaciju matematičkim izrazima". U ovom tekstu se bavimo problemom predstavljanja izraza, kao i osnovnim potprogramima za njegovu analizu.

Viktor Čerovski

Analizirajući strukturu i proces izračunavanja i najjednostavnijih izraza, poput **2+3*4** možemo uočiti nekoliko važnih stvari. Pre svega, da se izraz sastoji iz operacija i podataka (konstanti i promenljivih) nad kojima te operacija deluju. Da bi rezultat bio tačan, važan je redosled izračunavanja: prvo je potrebno pomnožiti **3** i **4**, pa tek zatim dodati **2**. Prosto kažemo da množenje ima veći prioritet od sabiranja.

PREFIKSI I POSTFIKSI

Zatim primećujemo da operacije ne moramo obavezno navoditi između njihovih argumenata (to jest u infiksnom obliku), već **3*4** možemo da zapišemo kao **mul(3,4)**, gde sada **mul** označava množenje brojeva u zagradi. Tada kažemo da je operacija zapisana u prefiksnom obliku. Možemo i ceo nas izraz da prevedemo u prefiksni oblik. Tada **2+3*4** postaje **add(2,mul(3,4))**. Na prvi pogled kao da smo sve napotrebno zakomplikovali. Ovakav zapis međutim ima važnu prednost: u njemu je eksplicitno navedeno kako se izraz računa, jer nam više nije potrebna dodatna informacija o prioritetu pojedinih operacija. Stoga je ovakav oblik zapisa pogodniji za računar.

Najzad, postoji i postfiksni oblik zadavanja izraza. Naš izraz bi tada bio predstavljen sa **(2,(3,4)mul)add** ili jednostavnije sa **2 3 4 * +**. Mnogi kompajleri složenije izraze prevode upravo u ovakav oblik. Prednost ovakvog zapisa je u jednostavnijem računanju izraza: potprogram za računanje bi prosto mogao da čita izraz sleva udesno i svaki broj na koji naiđe da pamti u posebnom nizu, a kada naiđe na operaciju, prosto uzme onoliko brojeva koje je poslednje stavio u niz, koliko data operacija zahteva i umesto ovih brojeva u nizu ostavi rezultat primene date operacije. Kada izraz bude pročitan, u nizu će ostati samo jedan podatak – vrednost izraza.

IZRAZI I NJIHOVO ZNAČENJE

Međutim, nas pre svega interesuje ne računanje vrednosti izraza, već sami izrazi. Šta je to izraz? Definicija izraza bi mogla da glasi ovako: svaka konstanta ili promenljiva je izraz. Ako je **a** izraz, onda je i **(a)** izraz. Ako je i **b** izraz, a **f** neka binarna operacija, onda je i **f(a,b)** takođe izraz.

Kako ova definicija praktično „radi"? Vrlo jednostavno. 3 i 4 su konstante, pa su i izrazi. **Mul** je binarna operacija, pa je i **mul(3,4)** izraz. Najzad, 2 je izraz i **add** je binarna operacija, pa je i **add(2,mul(3,4))** takođe izraz. Dakle, videmo da ovakva definicija obuhvata naš jednostavan primer. Ali, manje je očigledno da smo njom obuhvatili i praktično bilo koju drugu matematičku formulu.

Gotovo istovetno možemo da definišemo i drvo izraza – strukturu podataka kojom se izraz prirodno može predstaviti. Moderni proceduralni jezici, kao što su C i Paskal, imaju ugrađene sve potrebne mehanizme za kreiranje ovakvih struktura. O načinu formiranja i „obrade" dinamičkih struktura podataka je pisano u svakoj knjizi o C-u ili Paskalu, ali i u „Računarima", pa se nećemo time detaljnije baviti. Tim pre što nama ovako komplikovane strukture podataka neće trebati – zadovoljićemo se običnim nizom karaktera.

Od korisnika očekujemo da izraze zadaje u njemu najbližem, infiksnom obliku i to kao niz karaktera (to jest kao string). Zato najpre preciznije definišimo koji će nizovi karaktera biti izrazi. U tu svrhu izdvojmo prvo sve karaktere koji nam predstavljaju infiksne binarne operacije (to će biti simboli '+', '-', '*' itd.). Da bismo potprograme za analizu izraza pojednostavili, zahtevaćemo da sve operacije i funkcije budu predstavljene samo jednim karakterom. Takođe ćemo izdvojiti i male i srednje zagrade. Sada možemo da kažemo da je izraz bilo koji niz karaktera koji ne sadrži ove zagrade ili neki od karaktera koji predstavlja binarnu operaciju. Zatim, da ako je **a** neki izraz, to je i string '(+a+)', kao i '[+s+]'.

Ako je **o** neka od pomenutih binarnih operacija, a **u** i **v** su izrazi, tada je izraz i **u+o+v** i najzad, ako **f** nije binarna operacija ili neka od zagrada, izraz je string 'f+'(+a+)' kao i 'f+'[+s+]'.

Ono što pažljivijem čitaocu nije moglo da promakne u navedenoj definiciji izraza je da su sve operacije binarne (to jest imaju po dva argumenta) ili unarne (sa samo jednim argumentom). Šta, međutim, ako poželimo da uvedemo recimo funkciju **min** (koja daje vrednost najmanjeg od njenih argumenata) sa neodređenim brojem argumenata? Na primer, ravnopravno bi mogli da je upotrebimo kao **min(1,a,b,c)** ili **min(x,0)**. Posloji elegantno rešenje za ovaj problem, a to je da definišemo binarnu operaciju ',' (zarez) koju prosto shvatamo kao operaciju spajanja dva izraza u uređeni par. Sada nema nikakvih prepreka da i na primer string **a[x,y,z]** zadovolji našu definiciju izraza.

FUNKCIJA *ROOT*

Sledeći korak je pisanje procedure za obradu ovako definisanih izraza. U tome je ključna funkcija **root**. Ona iz zadatog izraza izdvaja onu operaciju koja ima najnizi prioritet pri računanju izraza – u slučaju našeg primera **2+3*4** to je operacija sabiranja. **root** takođe izdvaja i argumente te operacije (izraze **2** i **3 4**).

Da bi binarne operacije bile prepoznate, more biti definisana sa **newbiner** procedurom. Unarne operacije se zadaju sa **newunar** pro-



Listing 1

```
unit expr3;
{
                    Expression Manipulator
                        part 3b  v1.0
                Cerovski Viktor copyright XI 1991
}

INTERFACE

uses  SYMB1;

const
      der=#0;   NEG=#1;   INV=#2;
      BRA='(';  KET=')';

function  PAIR     (var s : expr; j : byte) : byte;
procedure CLEBRA   (var s : expr);
function  ROOT     (s : expr; var left, right : expr) : char;
function  CONS     (s : char; l, r : expr) : expr;
function  CONSTR   (s : char; l, r : expr) : expr;
function  XSUBDIV  (s:expr) : expr;

IMPLEMENTATION

function PAIR (var s:expr; j:byte) : byte;
var  m,n l:byte; f:char;
begin
    n:=0; m:=0; l:=length(s); f:=s[j];
    if f='(' then n:=1 else n:=-1;
    while ((m <> n) or (m=0)) and (j<l) do
    begin
        inc(j);
        case s[j] of
         '(' :inc(m);   ')' :dec(m);
         '[' :inc(m);   ']' :dec(m)
        end
    end;
    if ((m <> m)=0) or (pos(f,BRA)<>pos(s[j],KET))
       then writeln('Zagrade nisu dobro uparene!');
    PAIR:=j
end;

procedure CLEBRA (var s : expr);
var l  f : byte;
begin
    l:=length(s);
    while (pos(s[1],BRA)>0) and (PAIR(s,1)=l)
          do begin dec(l,2); s:=copy(s,2,l) end
end;

function ROOT (s : expr; var left, right : expr) : char;
var  l  j  l  prio : byte;
begin
    CLEBRA(s);
    l:=0; l:=length(s);
    for j:=1 to l do
    begin
        comm:=pos(s[j],binoper);
        if (comm>0) and ((l=0) or (prior[comm]<=prior[pos(s[l],binoper)]))
           then l:=j
           else if (pos(s[j],BRA)>0) then j:=PAIR(s,j)
    end;
    if l=0 then if (l>1) and (pos(s[1],NEG)>0)
               then begin ROOT:=s[1]; left:=copy(s,2,255); right:=''  end
               else begin ROOT:=NOP;  left:=s;              right:=''  end
           else begin
                  ROOT:=s[l]; left:=copy(s,1,l-1); right:=copy(s,l+1,255)
                end
end;

function CONS (s:char; l r:expr) : expr;
var  dummy : expr; c1, c2 : char;
begin
    if s>'  then CONS:='('+l+')'+s+'('+r+')'
         else if s=NOP then CONS:=s+'('+l+')'
                       else CONS:=l
end;

function CONSTR (s:char; l r:expr) : expr;
var  dummy s: expr;
      c1  c2 : char;
      p      : byte;
begin
    if s>'  then
    begin
        p :=prior[pos(s,binoper)];
        c1:=ROOT(l,dummy,s);
        if (c1>NOP)and(s>' ')and(prior[pos(c1,binoper)]<p) then l:='('+l+')';
        c2:=ROOT(r,dummy,s);
        if (c2>NOP)and(s>' ')and(prior[pos(c2,binoper)]<=p) then r:='('+r+')';
        CONSTR:=l+s+r
    end
    else if (pos(l[1],NEG)=0) or (PAIR(l,1)<>length(l))
           then CONSTR:=s+'('+l+')'
           else CONSTR:=s+l
end;

function XSUBDIV (s:expr) : expr;
var  l  r : expr; c : char;
begin
    c:=ROOT(s,l,r);
    if pos('-',l) or pos('/',l) > 0 then l:=XSUBDIV(l);
    if pos('-',r) or pos('/',r) > 0 then r:=XSUBDIV(r);
    if c>NOP
    then case c of
           '-' : XSUBDIV:=CONSTR('+',l,CONSTR(NEG,r,''));
           '/' : XSUBDIV:=CONSTR('*',l,CONSTR(INV,r,''));
          else XSUBDIV:=CONSTR(c,l,r)
          end
    else XSUBDIV:=s
end;

begin
```

Listing 2

```
Unit symb1;
{
                    Expression Manipulator
                        part 1a  v1.0
                Cerovski Viktor copyright XI 1991
}

INTERFACE

type  expr = string [255];

const
      MAXUNAR=10; MAXBINAR=10; MAXSYMB=20;
      delta=#127;

var   unaroper  :string[MAXUNAR];
      unardescr : array [1..MAXUNAR] of string[8];

      binoper   : string[MAXBINAR];
      bindescr  : array [1..MAXBINAR] of string[8];
      prior     : array [1..MAXBINAR] of byte;
      Abell     : array [1..MAXBINAR] of boolean;

      symb  : string[MAXSYMB];
      token : array [1..MAXSYMB] of string[15];

procedure NEWUNAR  (descr:string; code:char);
procedure NEWBIN   (descr:string; code:char; p:byte; comutat:boolean);
procedure NEWSYMB  (descr:string; tcode:char);
function  MACRO    (s, x, t : expr) : expr;
function  TOKENS   (s : expr) : expr;
function  UNTOKENIZ (s : expr) : expr;

IMPLEMENTATION

procedure NEWUNAR (descr:string; code:char);
begin
    unaroper:=unaroper+code;
    unardescr[length(unaroper)]:=descr
end;

procedure NEWBIN (descr:string; code:char; p:byte; comutat:boolean);
begin
    binoper:=binoper+code;
    bindescr[length(binoper)]:=descr;
    prior[length(binoper)]:=p;
    Abell[length(binoper)]:=comutat
end;

procedure NEWSYMB (descr:string; tcode:char);
begin
    symb:=symb+tcode;
    token[length(symb)]:=descr
end;

function MACRO (s, x, t : expr) : expr;
var d : string;

    l,n : byte;
begin
    d:=''; l:=length(s);
    repeat
        n:=pos(x,s);
        if n>0 then
        begin
            d:=d+copy(s,1,n-1)+t;
            s:=copy(s,n+l,255)
        end
    until n=0;
    MACRO:=d+s
end;

function TOKENIZ (s:expr) : expr;
var  l:byte; t:expr;
begin
    t:=s; s:='';
    for i:=1 to length(t)         do if t[i]<>' ' then s:=s+t[i];
    for i:=1 to length(symb)      do s := MACRO(s, token    [i], symb    [i]);
    for i:=1 to length(binoper)   do s := MACRO(s, bindescr [i], binoper [i]);
    for i:=1 to length(unaroper)  do s := MACRO(s, unardescr[i], unaroper[i]);
    TOKENIZ:=s
end;

function UNTOKENIZ (s:expr) : expr;
var l:byte;
begin
    for i:=1 to length(unaroper) do s:=MACRO(s,unaroper[i],unardescr[i]);
    UNTOKENIZ:=s
end;

begin
unaroper:='';  binoper:='';  symb:='';
NEWSYMB('sum',#228);   NEWSYMB('pi',#227);   NEWSYMB('delta',#235);
NEWSYMB('Delta',delta);NEWSYMB('begin',#174);NEWSYMB('end',#175);
NEWSYMB('alpha',#224); NEWSYMB('beta',#225); NEWSYMB('gamma',#226);
end

NEWUNAR('neg',NEG); NEWUNAR('inv',INV);
NEWBIN('^','^',70,false); NEWBIN('-','-',30,false); NEWBIN('+','+',40,true);
NEWBIN('*','*',40,false); NEWBIN('=','=',50,true);  NEWBIN('/','/',49,true);
NEWBIN('=','=',30,true);  NEWBIN('<','<',30,true);  NEWBIN('>','>',30,true);
end
```

Modul sa funkcijama *root* i *xsubdiv*

Osnovne funkcije za rastavljanje izraza

cedurom. Ove procedure kao prvi argument imaju ime operacije onakvo kakvo ga korisnik zadaje, a kao drugi argument koood kojim se data operacija predstavlja u izrazu. Definisanje binarne operacije zahteva zadavanje još dva parametra: prioriteta i informacije da li je operacija komutativna. Unarne operacije nije potrebno eksplicitno definisati, jer će **root** svaki

Listing 3

```
program ( Cvjetkovic Viktor (c) XI 1991 ) EXAMPLE1;

{$M 65500 0 655000}

uses Crt, STMB1, STMB2;

procedure showTree (e:expr; tab:string);
const dtab:string=   ';
var  l, r : expr; o : char;
begin
     if e=''  then exit;
     o:=ROOT(e,l,r);
     if o<>#0 then begin
                   writeln(tab,UNTOKENIZ(o));
                   writeln(tab,'L: '); showTree(l,tab+dtab);
                   if r<>''  then writeln(tab,'R: '); showTree(r,tab+dtab)
                end
             else if l<>''  then writeln(tab,l);
end;

var  e,l,r,s,t : expr; o:char;
begin
     clrscr;
     repeat
          write(#10,'Unesi izraz > '); readln(e);
          e:=TOKENIZ(e);
          showTree(e,'');
          writeln(UNTOKENIZ(XSUBDIV(e)))
     until e=''
end.
```

Primer upotrebe funkcija za rastavljanje matematičkih izraza

izraz oblika **f(..)** prepoznati kao primenu unarne operacije f nad argumentom u zagradi. Ako zadamo, na primer, izraz **a(x,y,z)**, njega možemo shvatiti kao „trinarnu" relaciju u prefiksnom obliku, ali i kao neku funkciju tri promenljive.

Ako već **root** „automatski" prepoznaje unarne operacije, čemu onda služi **newunar**? Problem je što sve operacije moraju biti opisane samo jednim karakterom, pa, na primer, **sin(x)** prema našoj definiciji nije izraz. Ali ako izvršimo **newunar**(sin chr(1)›3‹ program će interno da tekst sin‹D› zameniti sa karakterom **chr(1)** i time izvrši tzv. tokenizaciju stringa. Sam proces zamene jednog niza karaktera u stringu drugim izvodi **macro** funkcija. Njeni argumenti su izraz **e** (to jest string) u kom se sva pojavljivanja podstringa **x** zamenjuju stringom **t**. Tokenizaciju vrši procedura **tokeniz**. Zato ne treba zaboraviti da se pre štampanja izraza-rezultata izvršiti „detokenizacija", koja se obavlja funkcijom **untokeniz**.

Rad funkcije **root** je jednostavan: ona redom traži neku binarnu operaciju (svaka definisana binarna operacija se upisuje u niz **binoper**) u zadatom izrazu, pri čemu „preskače" delove izraza u zagradama. Ovo preskakanje vrši funkcija **pair** pronalazeći onu zatvorenu zagradu koja odgovara datoj otvorenoj. Pri tom se ne pravi razlika između malih i srednjih zagrada, izuzev što svakoj otvorenoj maloj zagradi mora da odgovara zatvorena mala zagrada (analogno važi i za srednje zagrade). **root** funkcija dalje pamti krajnju desnu binarnu operaciju najnižeg prioriteta koju vraća kao rezultat (pritom vraća i argumente ove operacije). Ako takve operacije nema, proverava se da li je izraz oblika **f(...)**, kada se kao rezultat izdvaja f (odnosno pronađena unarna operacija), a ako ni to nije ispunjeno, ceo izraz se tretira kao konstanta (pri čemu njena dužina ne mora biti jedan karakter), a kao pronađenu operaciju **root** vraća **chr(0)**. Konstanta NOP u programu je definisana upravo kao **chr(0)**.

Treba uočiti da bez **root** funkcije ne bi mogli da formiramo (**binarno**) drvo izraza, jer nam **root** uvek određuje operaciju – koren i izraze – „sinove". Upravo zbog toga principijelno nema potrebe formirati drvo, već je moguće izraz sve vreme čuvati praktično u obliku u kom je zadat. Svakako treba istaći prednosti formiranog drveta izraza: ako je potrebno da se program „kreće" po izrazu (na primer, pri njegovom transformisanju), brže je menjati vrednost jednog pokazivača nego izvršavati ceo **root** potprogram. S druge strane, pronalaženje svih pojavljivanja jedne promenljive i, na primer, njena zamena nekom brojnom vrednošću je znatno jednostavnije ako izraz čuvamo u obliku stringa. Jednostavnije je napraviti kopiju nekog izraza.

Najveći problem pri realizaciji ove ideje predstavlja način na koji *TurboPascal* radi sa stringovima. String je moguće produžavati i skraćivati, ali bez obzira koliko imao karaktera, on stvarno zauzima uvek fiksnu količinu memorije koja odgovara njegovoj maksimalnoj dužini. Kombinujući tu činjenicu, ograničenu veličinu steka i rekurzivne procedure sa petšest lokalnih stringova, kod složenijih izraza kao „rezultat" rada takvih procedura možemo očekivati jedino poruku „Out of stack!"

Kako izbeći ovaj potencijalno vrlo neprijatan problem? Ako uočimo da je podizraz predstavljen uvek neprekinutim delom celog izraza, možemo da podizraze zadajemo sa dva broja: jedan bi predstavljao poziciju prvog karaktera podizraza u izrazu, a drugi njegovu dužinu. Tako se sa samo dva broja može se preneti informacija o poziciji podizraza u izrazu. Dobre strane su što kompleksnost izraza više ne bi zavisila od ograničenja maksimalne dužine stringova na 255 karaktera i ograničenja veličine steka na 64 kilobajta, ali bi zato pristupanje podizrazima, njihova promena itd. moralo ići preko posebnih procedura, koje bi zakomplikovale i usporile program. Ovo je ujedno lepa prilika za upotrebu C++-a.

Zašto smo malopre spominjali baš rekurzivne procedure? Iz prostog razloga što su i sami izrazi rekurzivno definisani. Elementarna stvar pri pisanju rekurzivnih programa je određivanje (ili bar procena) veličine memorije potrebne za čuvanje svih lokalnih promenljivih i povratnih adresa. Lokalne promenljive se čuvaju na steku, pa je račun jednostavan: ukupnu dužinu lokalnih promenljivih izraženu u bajtovima pomnožimo sa brojem samopoziva date procedure u dubinu. Znati broj rekurzivnih poziva, kako taj broj zavisi od ulaznih podataka i koji ulazni podaci zahtevaju najdužu obradu i najviše samopoziva, u mnogome znači razumeti kako data rekurzivna procedura radi. Prirodno možemo da uvedemo i funkciju „inverznu" funkciji **root**. Date su dve takve funkcije: **cons** i **constr**. Obe vrše spajanje dva izraza (ili jednog) sa datom binarnom (ili unarnom) operacijom. Tako će **cons(+ , a+b , c+d)** dati **(a+b)+(c+d)**, dok **constr** za iste argumente proizvodi izraz **a+b+c+d**. Spajanje sa unarnom operacijom izvodi se zadavanjem praznog stringa kao trećeg argumenta. Razlika između **cons** i **constr** je što druga funkcija kontroliše i prioritete „korenih" operacija podizrazaza koje spaja i time izbegava pojavljivanje suvišnih zagrada kao u navedenom primeru.

FUNKCIJA *XSUBDIV*

Funkcija **xsubdiv** svako pojavljivanje oduzimanja (deljenja) prevodi u sabiranje sa negiranjem (množenje sa inverzijom). Na primer, izraz **a-b/c** transformiše u **a+neg(b inv(c))**. S obzirom da su sa, pored dve binarne operacije (koliko ima i polazni izraz) pojavile i dve unarne, dobijeni izraz je na prvi pogled nepotrebno komplikovaniji. U stvari smo postigli sledeće: dve binarne nekomutativne operacije („-" i „/") smo izrazili preko dve komutativne („+" i „*") i dve unarne. Značaj ovakve procedure biće jasniji kada se budemo intenzivnije bavili problemom transformisanja izraza. Zasad primetimo, s obzirom da pojedine operacije i simbole iz izraza možemo interpretirati na različite načine, da je izraz **a inv(b)** našto opštiji od izraza **a/b**, jer ako bismo **a** i **b** shvatili kao dve regularne matrice iz grupe svih regularnih matrica, tada prvi izraz lako interpretiramo kao množenje matrice **a** sa inverznom matricom od **b**, dok bi drugi izraz tada sadržavao operaciju deljenja dve matrice, što se ne definiše (bar to ne rade matematičari, fizičari i programeri).

Pozabavimo se detaljnije funkcijom **xsubdiv**, jer kao prva funkcija koja vrši neku transformaciju izraza predstavlja „prototip" za buduće transformacione funkcije. Njen rad možemo opisati ovako: sa **root** funkcijom se izraz rastavlja na dva podizraza. Ova podizraze ponovo povezuje **constr**. Ali ako je operacija koja ih povezuje (a dobijena sa **root**) oduzimanje ili deljenje, povezivanje se vrši saglasno transformacionom pravilu. U protivnom **constr** samo spaja ono što je funkcija **root** rastavila. Pri tom se na svaki podizraz rekurzivno primenjuje isti postupak, jer nije unapred jasno da li u podizrazu ima još operacija oduzimanja ili deljenja koje treba transformisati. Ovde dolazi do izražaja ideja da se izrazi čuvaju u obliku stringova, jer provera da li se u datom izrazu pojavljuju tražene operacije svodi se na proveru da li string sadrži karaktere - ili /. Da je izraz predstavljen drvetom, ovakvo pretraživanje ne bi bilo racionalno. Zato su u **xsubdiv** ugrađene provere pos funkcijama ne bi li se uštedeo poneki samopoziv.

xsubdiv ponekad mora da ispita svaki podizraz datog izraza (na primer kod **a/b-c/d-e/f**). Broj rekurzivnih poziva u dubinu u tom slučaju je jednak dubini drveta izraza. Funkcija ima tri lokalna stringa: **a**, **l** i **r** (promenljivu **o** i povratnu adresu, koje se takođe čuvaju na steku, zanemarujemo). Ako je dužina stringa kojim se predstavlja izraz, na primer, 200 karaktera i veličina steka 60 kilobajta, prostim računom zaključujemo da je moguće najviše 100 rekurzivnih poziva u dubinu, čemu odgovara izraz predstavljen drvetom dubine 100, što je za većinu praktičnih primena više nego dovoljno.

Pomenimo sada i jednu finesu vezanu za simbol -: **root** funkcija prepoznaje - kao binarnu operaciju, pa naizgled unarni minus ne može biti pravilno prepoznat. Međutim, ako bi sa **root** zahtevali korenu operaciju na primer izraza -a*f(x), rezultat će biti operacija -, desni podizraz **a*f(x)** a levi podizraz prazan string. Zato nadalje treba imati u vidu ovu specifičnost u prepoznavanju unarnog minusa.

Program *Example1*, kao što je i red, ilustruje upotrebu opisanih funkcija. Uneseni izrazi bi trebalo da budu pisani malim slovima. U programu je posebno važna direktiva kompajleru ($M) da poveća veličinu steka.

DINAMIČKA POLJA

Svako ko ima makar malo programerskog iskustva zna da rad sa višedimenzionalnim poljima ponekad ume da bude veoma nezgodan. U tom pogledu mnogobrojni nafrizirani C-prevodioci nisu nikakav izuzetak. Vešt programer će se, međutim, svemu dosetiti.

SAŠA MARKOVIĆ

Ako vam je u programu neophodno trodimenzionalno polje *f* realnih brojeva formata 3×4×5 moracete da ga deklarišete na sa mom početku sa

double f [3][4][5];

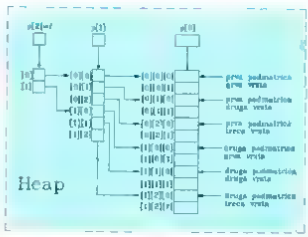
Ovakav pristup ima, pored svojih dobrih strana i brojna nedostatke. Alocirani memorijski prostor ostaje trajno zauzet sve dok (pol)program ne završi rad. Ako polje *f* zauzima trideset šest kilobajta a koristi se samo na jednom ili dva mesta u programu jasno je da ćemo, u velikom broju slučajeva raspoloživi memorijski prostor konstituiranacionalno — za podatke koje program permanentno koristi ostace manje RAM a. Drugi nedostatak je praktične prirode. Polje koje se deklarise na samom početku programa ima dimenzije odredene maksimalnim brojem podataka koje treba smestiti u polje. cak i ako je stvarni obim podataka daleko manji. Visak raspolozivog prostora ostaje trajno neiskoriščen. Takode, prenosenje ovakvih polja u potprograma predstavlja pravu moru (setice se loge svi oni koji su pisali programe za barabanje matricama u fortranu ili C-u). Teda se u potprogramu obavezno morsiju navesti i fizičke dimenzije polje kako bi kompajler mogao korektno da izvede interne kalkulacije sa ukazateljima. Ponekad je to korisno jer se lako mogu jednodimenzionalni nizovi u glavnom programu lako „prelormatirati" u visedimenzionalne matrice u potprogramu. No u vecini slucajeve to je pra nedostatak i izvor skrivenih grešaka nego prednost.

Sve je ovo posledica namere da se rad sa višedimenzionalnim poljima maksimalno pojednostavi. Elementima polja može se uvek lako pristupiti sa *f[i][j][k]* i od loga je teško smisliti nesto jednostavnije.

EFIKASNO REŠENJE

Naravno, veši programer će se svemu dosetiti, narocito ako je popul mene sa nepopravijivog i hermaticnog fortrana prešao na C. Rešenja se nameće samo: treba prihvatiti sve lepe osobina klasičnog nacina pristupa elementima polja, ali iz korena promeniti nacin deklarisanja. Evo uslova koje bi trebalo ispuniti za maksimalno komolan rad:

- umesto deklaracije polja *f* na samom početku programa treba dopustiti mogucnost reZervisanja memorijskog prostora na proizvoljnom mestu u programu
- elementima polja treba i dalje pristupati sa *f[i][j][k]* jer je to još uvek najrazumljivija i najsednostavnije.
- kada polje *f* vise nije neophodno treba omoguciti vracanje alociranog prostora programu
- pri prenošenju polja u potprogram u listi argumenata treba da bude naveden samo ukazatelj *f*. U samom potprogramu ne treba ponovo deklarisati fizičke dimenzije polja a nacin pristupanja elementima polja treba da ostane isti kao u glavnom programu
- potprogram koji treba da zameni klasični nacin deklarisanja visedimenzionalnih polja



Alokacija prostora u polju *f*

treba da bude univerzalan, odnosno da sa u najrazmenjenom obliku može koristiti za dvodimenzionalne kodimenzionalna ili *n*-dimenzionalna polja

Potprogram **dim**, čiji je listing priložen ispunjava sve navedene zahteve. Njegova deklaracija je sledeca

void dim (int s, int n, int lmax$_1$, int lmax$_2$, ..., int lmax$_n$)

Parametar ***s*** predstavlja veličinu (u bajtovima) jednog elementa polja. Vrednost ovog parametra najjednostavnije je utvrditi primenom ***sizeof()*** operatora. Parametar *n* predstavlja broj dimenzija polja i u potprogramu je ograničen na MAXDEPTH6. Konstante lmax, definišu fizičke dimenzije polja i njihov ukupan broj odgovara broju dimenzija *n*.

Upotreba ovog potprograma je krajnje jednostavna. Ako vam u programu zatraba polje tipa **VAŠ_TIP** dimanzija 5×5×2×3 postupak je sledeći

```
void main (void){
VAŠ_TIP ___f,
<C-naredbe>;
f=dim(sizeof(VAŠ_TIP), 4,5,5,2,3), /* bilo gde u programu */
<C-naredbe>;
f[i][j][k][l]=<po želji>;
<C-naredbe>;
dimfree(f,4), / bilo gde u programu /
<C-naredbe>;
}
```

Kao što se iz gornjeg primera vidi rezervisani prostor se oslobada pozivom potprograma ***dimfree (f,n)*** gde je *f* ukazatelj ne polje a *n* broj dimenzija polja.

PRINCIP REŠENJA

Sva snaga C-a počiva na ukazateljima. Bez njih bi ovaj jezik bio samo kriptična verzija bejzika III fortrana. Potprogram koji vam danas prezentiram počiva upravo na osnovnom pravilu ukazateljske aritmetike

f[i]=(f+i),*

Potprogram dim će prvo rezervisati prostor za smeštaj samih podataka korišćenjem sistemske funkcije calloc. Ukazatelj ne taj prostor biće *p[0]*. Zatim će biti rezervisan prostor za ukazatelje koji će definisati adrese u polju *p[0]*. Zatim će biti rezervisan prostor za ukazatelje koji će definisati adrese u polju *p[0]* na kojima pocinju kolone i taj prostor će imati ukazatelj *p[1]*. Nakon toga program rezervisa prostor za smeštenje ukazatelja koji će definisati adrese u polju *p[0]* na koj-ma počinju podmatrice i taj prostor će imati ukazatelj *p[2]*. Na kraju biće definisano polje ukazatelja *p[n-1]* u kome će biti definisane adrese u polju *p[0]* na kojima pocinju podpolja dimenzija *n-1*. Šema pristupanja elementima polja *f[2][3][2]* data je na sledećoj slici

Evo šta se dešava ako je potrebno pristupiti elementu polja *f[1][2][1]*. Naradbu u programu

f1][2][1]

kompajler će prevesti u oblik

((*(f+1)+2)+1)*

Pri tome će *(f+1) dati edresu prve podmatrice koja je u loku inicijalizacije polja smeštena u polju sa ukazateljem *p[2]*. Zatim *(*(f+1)+2) daje adrasu druge vrste u prvoj podmatrici koja je u fazi inicijalizacije smestena u polju sa ukazateljem *p[1]*. Na kraju *(*(*(f+1)+2)+1) deje zahtevanu vrednost smeštenu u polju sa ukazateljem *p[0]*

O SAMOM PROGRAMU

Dva opisana potprograma, ***dim*** i ***dimfree*** napisana su u ***TurboC***-u koriscenjem ***small***-memorijskog modela. Uz njih je priložen i kratki demonstracioni program koji ce koriscenjem potprograma ***dim*** prvo rezervisati prostor za polje tipa ***double*** formata 5×5×5×5. Zatim će elementima polja dodeliti vrednosti na uobičajeni način. Na kraju program štampa sve vrednosti polja pri cemu demonstracije radi vrednosti elementa polja odreduje poz vom potprograma ***value***. Obratite pažnju da se polje u potprogramu deklarše samo ukazateljem! Program se može lako prepravili tako da koristi ***farheap***, što samo potvrduje vac davno poznatu činjenicu da ce svaki koristan program morali da bude prepravljen da bi se mogao upotrebiti.

Ovakav pristup višedimenzionalnim poljima knje u sebi ozbiljan nedostatak: osim dinamičkog prostora potrebnog za smeštaj podataka treba alocirati i dodatni prostor za smestaj internih ukazateljskih polja. Zato treba dobro odmeriti vrlne i mane jednog ovakvog pristupa problemu. No, nadam se da se iz njega može dosta loga naucili i da ce mnogima moji potprogrami biti korisni. Ostalo je za ostala.

Nastavlja se na strani 63

NAGRADNA PREMETALJKA

Poslednja pitalica u 1992. godini očito je bila prilično jednostavna – u predvidenom roku primili smo priloge 87 autora, od kojih je čak 86 uspelo da sortira brojeve u dvadeset jednom potezu!

Podsetimo se pre svega, postavka. Prirodni brojevi 1-25 raspoređeni su na slučajan način kao na slici 1. U svakom potezu bilo koja dva broja mogu međusobno zamentiti mesta, pri čemu je cilj da se brojevi rasporede u rastući niz 1, 2, 3, ..., 25. To je naravno, trebalo uraditi primenjujući što manje poteza.

Posmatrajmo podnizove koje karakteriše da je na mestu A broj B, na mestu B broj C, ..., na mestu X broj A. Dobijaju se podnizovi

3-21-5-9-8-16-1
17-12-25-13-20-24-18-19-10-22-11-7-2
14-23-6-4
15

Svaki podniz dužine K se primenom K-1 zamene uređuje (u svakom „potezu" se jedan broj dovodi na pravo mesto, a poslednjom zamenom 2 broja) što se lako dokazuje indukcijom. Iz ovoga sledi da će poslojanje podniza smanjili potreban broj zamena za jedan. Da je ceo dati niz brojeva predstavljao jedan podniz bilo bi potrebno N-1. Ij. 24 premeštanja. Ovako, pošto ima 4 podniza, potrebno je 25-4=21 premeštanje. Redosled zamena dal je na slici 2, a do njega se može doći i primenom programa sa slike 3.

Posto su rešenja bila prilično slična sve nagrade smo izvukli. Tako je prvu nagradu (2400 dinara) dobio *Ranko Tomić* iz Gornjeg Milanovca, drugu (1800 dinara) *Ivan Malčević* iz Novog Beograda, a treću (1500 dinara) *Stjepan Šetula* iz Iloka. Specijalna nagrada za najinteresantnije rešenje dostavljeno modemom ovoga puta ide u Beograd – zaslužio ju je *Stojan Milorsdovic* (*smiloradovic* na Sezamu). Pohvale su zaslužili Računarska sekcija gimnazija iz Doboja (mentor *Mirko Boček*), *Biljana Srdanov, Valon Budima, Predrag Drljača, Ljubomir Josifovski, Boško Koprivica, Milovan Kovačević, Robert Pausic* i *Miloš Prvulović*.

REZULTATI PROŠLOGODIŠNJEG...

Došlo je vreme i za razultate upravo završenog (pelog po redu) godišnjeg takmičenja rešavača pitalica. U toku 1991. evidentirali smo 677 tačnih rešenja koje je poslalo 419 rešavača – odziv je, na žalost, manji nego prethodnih godina (primećuje se, osim toga, da je jako mnogo onih koji su rešavali samo jedan zadatak) a nema ostaje da razmislimo da li je to zbog preteških ili pratakih problema, skromnih nagrada, smanjenja tržišta ili nekog trećeg razloga. Malo opravdanje može da bude i to što je u 1991. godini izašao jedan broj „Računara" manje nego u 1990. No, taj ponešto smanjeni odziv niukoliko ne umanjuje vrednost brojnih priloge koje je ova rubrika primila, pa korisnimo priliku da čestitamo svim učesnicima takmičenja na pokazanom znanju i dobrim idejama koja su ugradili u svoja rešenja i programe.

Prilikom bodovanja rešenja primenjivali smo standardni kriterijum – svako korektno rešenje donosilo je po jedan poen, dva dodanla poena su se dobijala za

1	Slavko Kukrika Ljubljana	–	13++ poena
2	Boško Koprivica Sremčice	–	9+ poena
3	Milovan Kovačević, Sremska Kamenica	–	8+ poena
4	Miloš Prvulović Beograd	–	7++ poena
5	Predrag Cvetković Novi Beograd	–	6++ poena
6	Robert Paušić, Šid	–	6 poena
7	Andonov/Velkovski Skopje	–	4+ poena
8	Đorđe Ljubičić Beograd	–	4+ poena
9-10	Đorđe Rainović, Kisač	–	4 poena
9-10	Bratislav Veljković, Paraćin	–	4 poene

Rezultati godišnjeg takmičenja u rešavanju pitalice

72: D = _____

Ime i prezime _____

Adresa _____

Mesto _____

Kompjuter _____ Vreme _____

Identifikacioni broj: |_|_|_|_| |_|_| |_|_|

iz broja tel. | god. rod | po izboru

NOVOGODIŠNJI PROBLEM

Nikad nije kasno za pitalicu koja se zasniva na oznaci godine u koju polako „trošimo". Dal je broj A=1992[10] (broj 1992 podignut na stepen 1992). Neka je B zbir svih cifera broja A, C zbir cifara broja B a D zbir cifara broja C. Odrediti D.

Rešenja pitalice, zajedno sa **obrazloženim** programima koja sle napisali, pošaljite na uobičajenu adresu „Računari" (za Dejanove pitalice). Bulevar vojvode Mišića 17, Beograd. Iako da pristignu **pra** 25. februara 1992. Sva pisma sa korektnim rešenjima konkursu za novčana nagrade od 2400, 1800 i 1200 dinara, dok će kuponi (ili njihova fotokopije) na koje je upisan identifikacioni broj učestvovati u godišnjem takmičenju rešavača pitalice. Identifikacioni broj dobijete iako što u prve tri kućice upišete **poslednja** tri cifre nečijeg broja telefona u sledeće dve godinu vaseg rođenja i na kraju dodate dve cifre po izboru. Obratite pažnju da sva rešenja koja šaljete u loku godine nose isti identifikacioni broj. Osim običnom, rešenja ova pitalice možete da pošaljete i elektronskom poštom posredstvom Sazama – koristite menje REDAKCIJA PITALICE. REŠENJE ili iz komandnog moda: posaljite ličnu poruku korisniku pitalice (*mali wnte pitalice*). Sva rešenja prsligla elektronskom poštom ravnopravno sudeluju u izvlacenju nagrada i konkursu za **specijalnu nagradu** od 1500 dinara, koja će biti dodeljena najboljem rešenju koje je dostavljano modemom!

Slika 1

```
 3  17  21  14   9   4   2  10   8  22   7  29  20
23  19   1  12  19  10  24   5  11   6  18  12
```

Uredili slučajno rasporedene prirodne brojeve od 1 do 25 u rastuci niz u što manje poteza

Slika 2

```
3-21;  1-21; 18-21;  8-21;  9-21;  9-21; 17-12;
2-17;  7-12; 11-12; 22-12; 18-12; 13-12; 18-12;
24-12; 20-12; 12-12; 25-12; 14-23;  4-23;  6-22;
```

Redosled zamena

Najbolji u 1991. godini

osvojenu prvu nagradu a po jedan dodatni poen za drugu nagradu ili nagradu za najbolje rešenje dostavljeno modemom. Pri jednakom broju poena više mesto pripada onome ko je osvojio više nagrada (plus u tabeli) a ukoliko je i to isto onome ko je osvojio više prvih nagrada. I pored ih detaljnih pravila 9. i 10. mesto ove godine delе dva rešavača a rang listu dajemo u posebnoj tabeli.

Godišnja nagrada od 12.000 dinara dakle pripada **Slavku Kukriki** koji je uspešno rešio devet od deset prošlogodišnjih pitalica i osvojio dve prve (u martu i junu 1991. godine) i jednu treću nagradu. Gospodin Kukrika je, ujedno, prvi kome je uspelo da osvoji čak dve godišnja nagrada za rešavanje pitalica – 1990. i 1991. godine. Čestitke dakle, treba duplirati! A posebno nam ja drago što su sva rešenja stigla modemom preko Sezama.

Drugo mesto zauzeo je *Boško Koprivica* iz Sremčice koji je lakođe rešio devet pitalica i osvojio treću nagradu u oktobru 1991. Gospodin Koprivica je učestvovao i u našem prethodnom takmičenju kada je osvojio 9. mesto. Na trecem mestu (8 poena) je *Milovan Kovačevic* iz Kamenica koji je uspešno rešio sedam pitalica i osvojio drugu nagradu u junu 1991. – podsetimo se da je gospodin Kovačević već osvajao drugo mesto u našem prvom godišnjem takmičenju, lokom 1987. godine. Čestitke pripadaju i svima ostalima sa liste „deset naj" a naravno i onima koji su, slicajem okolnosti, ostali „ispod crte". Ovo je i dobre prilika da se podsetimo na pobednike naših prethodnih takmičenja 1990. ja rekosmo prvi bio *Slavko Kukrika* iz Ljubljane, drugi *Đorđe Rainović* iz Kisača a treći *Bratislav Veljković* iz Paraćina. 1989. ja prvo mesto osvojio

Slika 3

```
program permutacije;

uses crt;

type matr = array [1..30] of integer;

var c: char;
    spec: string[60];
    sbr: string[2];
    a: matr;
    i, j, br: integer;

procedure zamena (var a: matr; x,y:integer; var br: integer);
var pomoc: integer;
begin
   pomoc:=a[x];
   a[x]:=a[y];
   a[y]:=pomoc;
   inc(br);
end;

begin
   spec:= 031721140504026002207252023150121910240511061013 ;
   for i:=1 to 25 do
   begin
      sbr:=copy(spec, 2*(i-1)+1, 2);
      val (sbr, a[i], j);
   end;
   br:=0;
   i:=1;
   while i<25 do
   begin
      if (a[i]<>i) then zamena (a, i, a[i], br);
      i:=1;
      while i=a[i] do inc(i);
   end;
   for i:=1 to 25 do write(a[i], ' ');
   writeln;
   writeln( Broj zamena je , br);
end
```

Program koji vodi do rešenja

Jadran Krašovec iz Kozine, drugi je bio *Đorđe Ljubičić* iz Beograda a treći *Slavko Kuknka*. 1988. godine je pobedio *Milomir Aleksić*, drugi je bio *Leo Bosnić* a treći *Dragan Grbić*. Godinu dana pre toga, 1987., pobedio je *Milan Grbić*, drugi je bio *Milovan Kovačević*, a treće mesto je podelio veći broj rešavača. Svima onima koji već godinama sa velikim uspehom rešavaju naše nagradne zadatke, kao i onime koji će ovoga meseca po prvi put okušavati svoje znanje i svoj kompjuter, želimo mnogo uspeha i potrebnu količinu „sportske sreće" u 1992. godini.

...I PRAVILA ZA OVOGODIŠNJE TAKMIČENJE

Zbog pomeranja rokova predaje u štampu rukopisa za „Računare" u prošlom broju nismo objavili propozicije našeg šestog godišnjeg takmičenja rešavača pitalice. Takmičenje je počelo sedamdeset prvom pitalicom (u prošlim „Računarima") i trajaće tokom čitava 1992. godine. Sva rešenja koja primimo konkurišu za redovne nagrade, dok će pisma na koje je upisan i identifikacioni broj konkurisati i za specijalnu godišnju nagradu. Ova će nagrada pripasti čitaocu „Računara" koji u toku 1992. godine osvoji najviše poena; po jedan se poen dobija za svako korektno rešenje pitalice, dok će prvo- i drugonagrađeno rešenje, kao i najodlje rešenja poslato modemom, dobijati dodatna dva odnosno po jedan poen. Samo se po sebi razume da svaki čitalac može da pošalje samo po jedan odgovor mesečno – duplirani će odgovori eventualno ulaziti u redovna izvlačenja, ali neće donositi dodatne poena!

Ostalo je još da ponovimo recept za određivanje identifikacionog broja. Setite se najpre svog broja telefona (ako nemate telefon, broja telefona nekog prijatelja) pa izdvojite njegove tri **poslednje** cifre. Na to dopišite dva cifre koja predstavljaju godinu vašeg rođenja a zatim još dve cifre po izboru. Naš Sezam bi, na primer, mogao da ima identifikacioni broj 899 89 10 jer mu je broj telefona 648-899, jer je osnovan 1989. godine i jer u trenutku pisanja ovog teksta radi na 10 linija. Identifikacioni broj upisujete u predviđene kućice našeg kupona a zatim ovaj kupon prepisujete ili lepite na prvu stranicu rešenja. Vrlo je bitno da sva rešenja koja pošaljete nose isti identifikacioni broj, u protivnom možete da ispustite godišnju nagradu!

Za kraj smo ostavili jedno upozorenje proisteklo iz višegodišnje prakse: identifikacioni broj je formiran na način koji praktično isključuje ponavljanja, pa za ovih dvadesetak meseci nismo pronašli dva jednaka „korektna" broja. Neki rešavači su, međutim, želeli da budu originalni pa su birali brojeve poput 000 00 00, 313 13 13 ili 011 11 11 – ako je vama neki ovako „originalan" broj „pao na pamet", budite sigurni da je još nekome, a to pravi probleme, pre svega rešavačima čiji odgovori bivaju zanemareni. Zato vas molimo da ove godine izaberete broj prema izloženim pravilima – to će vas najbolje zaštititi od dupliranja! Vaš novi identifikacioni broj, jasno, može ali i ne mora da bude jednak starom.

Nastavak sa strane 61

```
/**************************************************************************\
*                                                                          *
*           MULTIDIMENSIONAL RUN TIME ARRAYS FOR TURBOC                    *
*                                                                          *
*                    [C] Markovic Sasa 1991                                *
*                                                                          *
*  YOUR TYPE ***...**f = dimalloc(YOUR_TYPE) n imax  imax      imax )      *
*            \_____/                            1      2          n        *
*            n times                                                       *
*                                                                          *
*  Also that you may use [i ][i ]...[i ] to acces array elements           *
*                          1   2      n                                    *
*                                                                          *
*  Use dimfree(f, n) to release allocated space                            *
\**************************************************************************/

#include <stdarg.h>
#include <stdio.h>
#include <stdlib.h>

#define MAXDEPTH  8

/* ALLOCATING SPACE */
void *dimalloc(... [illegible]
   int i, j, k, size[MAXDEPTH];
   unsigned [illegible] total2;
   void **p[MAXDEPTH];
   va_list argptr;
   va_start(argptr, n);
   if(n>MAXDEPTH) {
      printf("Too many dimensions specified \n");
      exit(1);
      }
   for(i=0; i<n; i++) {
      size[i]=va_arg(argptr, int);
      total*=size[i];
      }
   total2=total;
   for( [illegible] ) {
      p[k]=(void **)calloc(total, [illegible] (void **)) [illegible]
      if(p[k]==NULL) {
         printf("Cannot allocate heap space for %s \n
                 [illegible] pointers   Data );
         exit(1);
         }
      total/=size[j];
      }
   for( [illegible] ) {
      for( [illegible] total2/size[j]; k<total2; k++)
         p[i][k]=(void *)p[ [illegible] ];
         if [illegible]
            p[ [illegible] ]=(void**) [illegible]
         else
            p[ [illegible] ]=size[j];
      }
   va_end(argptr);
   return(p[0]);
   }

/* RELEASING SPACE */
void dimfree(void *pi, int n) {
   int i;
   void **p[MAXDEPTH];
   p[0]=(void**)pi;

   for(i=0; i<n [illegible] )
      p[i+1]=(void **)p[i][0];
   for(i=0; i<n; i++)
      free(p[i]);
   }

/* EXAMPLE */
void main(void) {
   double ****a;
   double value(double ****a, int i, int j, int k);
   int i, j, k, l;
   a=dimalloc(sizeof(double), 4, 5, 5, 5, 5);
   for(i=0; i<5; i++)
      for(j=0; j<5; j++)
         for(k=0; k<5; k++)
            for(l=0; l<5; l++)
               a[i][j][k][l]=i+2*j+3*k+4*l+0 [illegible]
   printf("\n");
   do {
      printf("4 indeksa   ");
      scanf("%d%d%d%d", &i, &j, &k, &l);
      printf("a[%d][%d][%d][%d]=%f\nProvera   %f\n", i, j, k, l,
         value(a, i, j, k, l), [illegible] );
      } while(i!=[illegible]);
   dimfree(a, 4);
   exit(0);
   }

/* IT IS SIMPLE TO USE UP ARRAYS IN SUBROUTINES */
double value(double ****array, int i, int j, int k, int l) {
   return array[i][j][k][l];
   }
```

Dinamičko odvajanje prostora za višedimenzionalne nizove

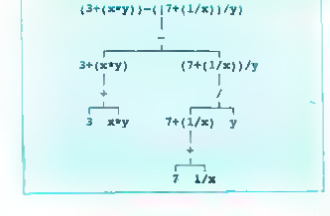
U sasvim slobodnom tumačenju, za rekurziju se može reći da je to slučaj kada neka procedura ili funkcija poziva samu sebe. Posto se ona zatim ponaša kao da ranije nije bila inicirana već kao potpuno nova funkcija/procedura (videćemo da to zapravo i jeste), pogrešne lokalne definisane promenljive i podaci se ne gube nego ponovo važe nakon povratka iz rekurzivnog poziva. Ovo je izuzetno korisno kada je u nekom cikličnom procesu potrebno stalno pamtiti i upoređivati rezultate odnosno kada se zeli realizovati tok izvršavanja programa prema strukturama liste i stabla.

REKURZIJA UOPŠTE...

Evo odmah konkretnog primera. U prvom naslavku je bio dat program DIJAGRAM za iscrtavanje grafika dala matematičke funkcije, ona je bila definisana unutar samog listinga u okviru funkcije **f(x y)** pa se shodno tome po pravođenju nije mogla menjati. Za korisnika tog programa bi zato bilo praktičnije da se funkcija može zadavati pri svakom novom startovanju na upit

Unesite funkciju: (3+(x*y))-((7+(1/x))/y)

Celi uneseni aritmetički izraz bi program primio kao alfanumerički parametar i u toku rada bi morao da ga raščlanjuje zamenjuje vrednosti za **x** i **y** i izračunava **z** koordinate. Problem je u sledećem: iduci sleva na desno i izvršavajući matematičke operande iim redom ne poštujući zagrade dobice će se pogrešni rezultati. Jer u navedenom izrazu konkretno neophodno je prvo izmnožiti **x** i **y**. To dodati trojci zapamtiti rezultat naći **1/x**, dodati sedmici podeliti sve sa **y** i konačno, to oduzeti od prethodnog rezultata. Celi provedeni postupak može se predstaviti u vidu jednog binarnog stabla



Da bi se aritmetički izrazi slični ovome mogli izračunavati treba imati rekurzivnu funkciju koja ih raslavlja na prosta sekvenca istovremeno izvršavajući operacije u njima. Takva funkcija bi primala argument tipa **string**, a vraćala bi tipnu vrednost kao **real**. Algoritam bismo mogli opisati na sledeći način (podrazumeva se da je izraz sintaksno ispravan):

function Vrlzraza(Izraz string):real

- ako string sadrzi samo jedan broj. Funkciji dodati njegovu vrednost i ići na kraj
- sve dok ne dobijaš znak za računsku operaciju koji nije unutar zagrada uklanjaj po par zagrada sa početka i kraja izraza
- podeli string izraza na dva dela: **Izraz1** od početka stringa do znaka **Izraz2** od znaka do kraja stringa
- dodeli sledeću vrednost funkciji

Vrlzraza =

Vrlzraza(Izraz1) mat operacija Vrlzraza(Izraz2),

end.

Celi primer je dat da bi se istakla poanta rekurzije – lokalne varijable bilo prosloj bilo slozenog tipa ostaju neizmenjene nakon rekurzivnog poziva. To je omogućeno činjenicom da polprogrami uvak daju pradnosi sopstvenim varijablama nad istoimenim i istoliipnim, bez obzira da li su ove deklarisane kao globalne kao lokalne u neko drugoj funkciji/proceduri, ili kao lokalne u istoj proceduri koje kasnije poziva samu seba.

Ako iz glavnog programa pokrenamo funkciju **Vrlzraza**, pa onda ona sa svoje strane poziva samu sebe pa ta nova opet učini isto itd, svaka od njih će čuvati sopstvene dalove slninga **Izraz** na različitim mestima u memoriji. Ta masta su pristupacna samo za tu funkciju koja ih je rezervisala, sve dok je ona aktivna. Znajući da bilo koja i bilo kakva funkcija ili procedura ne može da okonča sopstveno izvršavanje pre nego što lo učine sve ostala koja je ona ispozivala, zaključujemo da pravilo kretanja kroz lok programa pa tako i kroz rekurziju glasi „Last-In-First-Out" – isto kao i kod steka. Zapravo se argumenti potprogramima i prosleđuju putem sistemskog steka racunara.

U literaturi se redovno sreće zaključak da je rekurziju najbolje izbegavati ali on među lim najčešće sledi nakon nabrajanja čtjavog niza u sustini nerakurzivnih problema na primer izracunavanja faktorijala zadatog prirodnog broja. Rekurzijom se mogu napisati znatno krace i elegantnije rutine za mnoge trivijalne stvari koje bi svaki programer realizovao repeticijom ili iteracijom ali je ispravno da se rekurzija koristi jedino u resavanju prirodno rekurzivnih problema.

...I U TURBOPASCAL-U

Sve što je dosad rečeno važi za svaki jezik u kome je rekurzija moguca. Da vidimo kako sa tim stojimo u našem Turbo Pascalu.

Parametri koji se prosleduju funkcijama i procedurama poslavljaju se na slek redom kojim su deklarisani. I to tako da se prosti tipovi podataka (**integer real char boolean pointer**) upisuju direktno a slozeni (**array string record** i slicno) preko pokazivaca na memorijsku lokaciju sem ako nisu duzine do čalri baj ta. I u jednom i u drugom slučaju pozvani podprogram pravi kopije prosledenih parametara i sa njima dalje racuna kao sa sopstvenim lokalno deklarisanim varijablama. Na taj način na može doći do zamene sa varijablama identičnog identifikatora i lipa deklarisanim negda drugde.

Često se rekurzivni polprogram slavlja ugnježden unutar jednog spoljnjeg. To oluda što sve parametre koji sa prosleduju nije uvek neophodno kopirati jer se u toku rada ne menjaju, dok je sa druga strane ponekad potrebno sačuvati vrednosti nekih internih varijabli (recimo brojača u petlji) pa se i oni slavljaju u listu parametara. Pošto procedura ili funkcija koja je ugnjezdena u neku drugu može koristiti njene lokalne konstanta i promenljive nema potrebe da se u okviru rekurzivnog polprograma one deklarišu sa **var**.

Karakteristična *run time* greška pri rekurziji je prekoracivanje dozvoljena granice steka. Javlja se u slučaju kada programer ispusti ispitivanje uslova za rekurzivni poziv ili kada šalje preveliki broj parametara. Ako je program ispravan a iskače zbog **StackOverflow** onda je sam prislup bio pogresan – rekurziju narne valja izbegavati kada se ne zna tačno najveći broj pozivanja. Tipično se programu zadaje da nešto izrači a on to izrazi i izrači i crkne pre nego sto nade. Tada moramo ili da izmislimo pametniji algoritam ili da pomenmo maviše granicu steka – direktivom $M pa isko to mogne dobro je ako ne. Nalazi li se vaš rekurzivni polprogram u okviru zasebnog modula moracala ovu direklivu staviti u početak glavnog programa, pošto on sa svojim programskim jedinicama deli zajednicki slek i heap – selite se loga kada vam vaša rekurzivna procedura nakon sklapanja cele aplikacije stane iskakati zbog prekoračenja steka. Iako je ranije odlično funkcionisala. Inače dobar primer naskanlne upotrebe rekurzija su rutine za popunjavanje zatvorene oblasti na ekranu, moguće je osmisliti čak i takve likove za čije bi popunjavanje trebali magabajti steka!

TRI PRIMERA

Na listingu 1 dat je jednostavan programčić R_FRACTAL iz centra ekrana povuče se radijalno **gmax** linija dužine **d**, zatim se iz kraja svake od njih povuče opet **gmax** linija skracanih odnosno produzenih za faktor **q**, isti postupak se ponavlja ukupno **lmax** puta. Pored toga što generiše dopadljive slićice (probajte sa vrednostima 2 12 100 0 7 3 8 80 0 6 6,4 80 0 5), program sadrzi primer rekurzivna procedure gde su unapred zadani i stalni razgranalosti i dubina slabla.

Primer rekurzivne procedura u kojoj se razgranalosti i dubina slabla prilagodavaju problemu koji se resava dat je na listingu 2. Rec je o klasičnom primeru za rekurziju FILL rutini. Služi za popunjavanja zatvorene konture zadatom bojom praktično boje pozadine u programu **BojaPop**i **BojaPoz** respektivno. Ppounjavanje počinje iz zadane tačke **x y** i odvija se na sledeci način: ide se prvo ulevo sve dok se na nalelt na piksal različite boje od **BojaPoz** pa zatim isto lako udesno ove dve krajnje tačke se spoje linijom **BojaPop**. Sledi ispitivanja svih tačaka neposredno ispod i iznad linije i čim se nalda na praznu izvršava se rekurzivni poziv. Kada se ispitaju sve lačke i izvrsi rekurzivnih poziva koliko je potrebno (mnogo manje nego što je lačaka koje se ispituju – zasto?) kontrola se vraća proceduri niza ili glavnom bloku ako je posao zavrsen. Dve umelnule **Sound** naredba docaraće nam kretanje kroz rekurziju zvukom piskavi ćemo čuli pr popunjavanju. I kada sa pozivaju nove i nove FIJI procedure i parameli služu na slek dok ce se dubok zvuk javljati pri ispilivanju znači vracanju unazad i praznjenju sleka. I da bi sve bilo još efektnije u igri su još i tranutne y vrednosti koordinata!

Primetile da je moguće primenili i dalako kvalitetniji algoritam – do rekurzivnog poziva bi trebalo doci tek kada se sve tačke koje ispitujemo ne mogu povezati jednom linijom slo se javlja pri ispunjavanju konveksnih (ulubljenih) kontura. Ovako nas program jer smo hlali da buda što prostiji harći memoriju pa mu za „filovanje" tole složenijeg lika kao što je niz koncentričnih lukova treba staviti na raspolaganje celokupni slek. Inače prijavljuje **StackOverflow**. To je učinjeno na početku listinga direktivom $M 65520 0 655360 kojom smo p oširili valicinu segmenta steka sa predelin sanih 16384 bajtova na skoro celi segment. (Druge dve vrednosti za minimalnu i maksimalnu vall činu heap memorije nismo dirali.)

Pored grafike rekurzija je našla veliku primenu i u izradi matematičkog softvera. Prvo sto matematika sama po sebi obiluje rekurzivnim formulama a drugo što se mnoge melode kojima se s lakocom služa ljudi pri resavanju problema ne mogu uspešno implementirali na kompjuteri pa se pribegava primeni „sile" – isprobaju se sve moguće varijacije pa one koje

Listing 1 R_FRACTAL

```
{$M 65520, 0, 655360}

program R_FILL; { M.Olenjin; Oktobar 1991 }

uses Crt, Graph;

var Gd, Gm, x, y, x1, x2, n : integer;
    Sot, Vis, Luk           : word;

procedure Fill (x,y:integer; BojaPop,BojaPoz:word);
  procedure Popuni (x,y,n:integer);
  begin
    Sound(y+1000);
    if GetPixel(x,y) = BojaPoz then begin
      x1:=x; x2:=x;
      while GetPixel(x1-1,y) = BojaPoz do Dec(x1);
      while GetPixel(x2+1,y) = BojaPoz do Inc(x2);
      Line(x1,y,x2,y);
      for n:=x1 to x2 do begin
        if GetPixel(n,y-1) = BojaPoz then Popuni(n,y-1,0);
        if GetPixel(n,y+1) = BojaPoz then Popuni(n,y+1,0);
      end;
    end;
    Sound(y);
  end;      {Popuni}
begin
  SetColor(BojaPop);
  Popuni(x,y,0);
  NoSound;
end;

begin  {program}
  DetectGraph(Gd,Gm);
  InitGraph(Gd,Gm,'');
  Sot:=GetMaxX div 2;
  Vis:=GetMaxY div 2;
  Randomize;
  for n:=4 to 18 do begin
    Luk:=Random(360);
    Arc (Sot,Vis,Luk,Luk-30,n*(Vis div 18));
  end;
  Circle (Sot,Vis,(n+1)*(Vis div 18));
  Fill (Sot,Vis,1,0);
  repeat until KeyPressed;
  CloseGraph
end.   {program}
```

Rekurzivna procedura sa unapred zadatom i stalnom razgranatošću i dubinom stabla

Listing 2 FILL

```
program R_FRACTAL;   { M.Olenjin (c) Decembar 1991 }

uses Crt, Graph;

var Gd, Gm, x2, y2, gmax, imax : integer;
    d, q, k                    : real;

procedure Fractal (x1,y1,i,g:integer; d,ugao:real);
begin
  for g:=1 to gmax do begin
    Ugao:=Ugao+k;
    x2:=x1+round(d*sin(Ugao));
    y2:=y1+round(d*cos(Ugao));
    SetColor (i); Line (x1, y1, x2, y2);
    if i<imax then Fractal (x2, y2, i+1, g, d*q, Ugao+k/2);
  end;
end;

begin  {program}
  ClrScr;
  Window (20, 10, 60, 15);
  Write ('Unesite vrednost za broj grana     : '); Readln (gmax);
  Write ('Unesite vrednost za broj cvorova   : '); Readln (imax);
  Write ('Unesite vrednost za duzinu prve    : '); Readln (d);
  Write ('Unesite koeficijent progresije     : '); Readln (q);
  k:=2*Pi/gmax;
  DetectGraph (Gd, Gm);
  InitGraph(Gd, Gm, '');
  Fractal (GetMaxX div 2, GetMaxY div 2, 1, 0, d, -k/2);
  repeat until KeyPressed;
  CloseGraph;
end.   {program}
```

Rekurzivna procedura u kojoj se razgranatost i dubina stabla prilagođavaju problemu koji se rešava

Listing 3

```
program R_DETERMINANTE; { M.Olenjin; Decembar 1991 }

uses Crt;

var Det            : array [1..10, 1..10] of real;
    Vrsta, Kolona  : array [1..10] of Boolean;
    D2             : array [1..4] of word;
    RedMax,k1,k2   : integer;

function VrDet (Red,i,j,Kof:integer; Vrednost:real) : real;
begin
  if Red=2 then begin
    k1:=1; k2:=2;
    for i:=1 to RedMax do begin
      if Vrsta[i]  then begin D2[k1]:=i; Inc(k1); end;
      if Kolona[i] then begin D2[k2]:=i; Inc(k2); end
    end;
    Vrednost:=(Det[D2[3],D2[1]] * Det[D2[4],D2[2]])
             -(Det[D2[2],D2[3]] * Det[D2[4],D2[1]]);
  end
  else begin
    Vrsta[i]:=false;
    for j:=1 to RedMax do begin
      if Kolona[j] then begin
        if Det[j,i] <> 0.00 then begin
          Kolona[j]:=false;
          Vrednost:=Vrednost+(Kof*Det[j,i]*VrDet(Red-1,i+1,1,1,0));
          Kolona[j]:=true;
        end;
        Kof:=-Kof;
      end;
    end;
    Vrsta[i]:=true;
  end;
  VrDet:=Vrednost;
end;

begin  {program}
  ClrScr;
  Write('Unesite red determinante: '); Readln(RedMax);
  for k2:=1 to RedMax do begin
    for k1:=1 to RedMax do begin
      GotoXY(k1*8+(37-(RedMax*3)),k2+2);
      Read(Det[k1,k2]);
    end;
    Vrsta[k2]:=true; Kolona[k2]:=true; Writeln;
  end;
  Writeln('Determinanta iznosi: ', VrDet (RedMax, 1, 1, 1, 0));
  repeat until KeyPressed;
end.   {program}
```

Rekurzivnu funkciju *VrDet* koja unetu determinantu (najviše desetog reda) razvija po prvoj vrsti

odgovaraju postavci zadatka predstavljaju rešenja i gotov posao! Od značajnijih matematičkih problema za koje koristimo rekurziju mogu se pomenuti najmanji zajednički sadržalac, Fibonacijev niz i kombinatorika.

Kod traženja determinante neke kvadratne matrice se nećemo mnogo namučiti ako je ona drugog ili trećeg reda – samo oduzmemo proizvode dijametralnih članova, odnosno primenimo Sarusovo pravilo ili pravilo trougla. Determinante četvrtog i višeg reda moramo, pak, da rastavljamo na subdeterminante sve dok ne dođemo do 3×pa njih da izračunavamo, množimo sa kofaktorima itd. Kako je, dakle, razvijanje determinanti po prirodi rekurzivno, nije teško napisati program koji će to raditi umesto nas: popul ovoga sa trećeg listinga. On sadrži rekurzivnu funkciju **VrDet** koja unetu determinantu (najviše desetog reda) razvija uvek po prvoj vrsti: ako je ona drugog reda odmah vraća rezultat, a ako nije onda **VrDet** u odgovarajućim nizovima označi vrstu i kolonu članova koje ne treba da uzima u obzir pri daljem razvijanju subdeterminanti i zatim dalje pita samu sebe za njihovu vrednost.

Razvijanje determinante je moguće značajno ubrzati "kresanjem" stabla pozivanja: potrebno je pre rekurzivnog poziva odabrati prvi element vrste po kojoj razvijamo, koji nije jednak nuli, a potom elementarnim transformacijama nad kolonama svesti sve ostale elemente te vrste na nule. Tada je dovoljno tražiti vrednost samo jednog kofaktora, što će rezultovati stablom bez grana, tj. listom. Takav pristup ipak nije preporučljiv zbog dva razloga: kako se pri racionalisanju determinante koristi deljenje i kako se računar služi decimalnim brojevima a ne razlomcima, javiće se izvesno gubljenje na tačnosti, sem toga, pri deljenju vrlo velikih brojeva sa brojevima bliskim nuli može doći i do prekoračenja opsega računanja.

* * *

Rekurzija je, kao što smo videli, jedna izuzetno moćna tehnika – onaj ko je dobro savlada već može sebe da nazove programerom! Samo, da ne bude zabune, ne postoji nijedan problem koji je rešiv, odnosno nit jedan algoritam koji je ostvariv samo i jedino uz pomoć rekurzije. Moguće je skladištenje podataka i tok izvršavanja prema LIFO strukturi organizovati na više načina, i lako da neki prosti ili složeni niz proglasimo za stek, neku promenljivu za **StackPointer**, da napišemo **Push** i **Pop** procedure i slično. No, upravo je rekurzija i izmišljena da se u paskalu, moduli ili C-u ne bakćemo sa svim tim, već da u elegantnim jezicima pišemo i elegantne programe.

Naravno, ovaj tekst nije ni imao za cilj da izloži sve o ovoj interesantnoj temi. Zato ćemo se u jednom od narednih brojeva vratiti na nju i obraditi specijalne slučajeve rekurzija u Turbo Pascalu kao što su korutine, objasniti primenu rekurzija u pisanju logičkih igara i dati primer jednog jačeg programa gde se ona pojavljuje u punom sjaju. Do tada, pratite našu rubriku, posebno sledeći nastavak, u kome pišemo o bojama.

EOF

Gledajući sadržaj prethodnih napisa u ovoj rubrici, moram priznati da sam sa iznenađenjem otkrio da je iza nas lančno dvanaest nastavaka. Iako bih dvanaest nastavaka pokrivalo više od godinu dana, ipak sam imao utisak da se radi o znatno kraćem periodu – nekako je brzo prošlo.

Sa vase strane uglavnom nije bilo reakcija. Ako je to zato što ste ovu rubriku primili kao stalnog dobrodošlog i nenametljivog savetnika u osvajanju C-a, onda sam zadovoljan. Ako je moja procena pogrešna, ovo je dobar trenutak da mi kažete i da zajedno usmerimo ovu „školu" u pravcu koji vam više odgovara.

ENUM

Jedna od osnova dobrog pisanja (ma šta to bilo) mada je čitkost koda s gumo jedna od slavki) u C-u je maksimalno korišćenje usluga **prelprocesora**. C jezik odlikuje visoka prenosivost izvornog koda na najrazličitije platforme, ali je ovo samo delimično zasluga samog jezika. Deleko veći praktični značaj ima prelprocesor koji svojim **#define** i **#if** direktivama omogućava da se u jednom izvornom kodu, zapravo nađe nekoliko verzija pojedinih delova programa – delovi se potom aktiviraju pri kompajliranju promenom samo jedne **#define** direktive.

Jedan od naknadno ubačenih elemata C-a koji se ne koristi previše, ali lakode dopunjuje čitljivosti programa je i **ENUM** tip podataka. Radi se o tipu kod koga se unapred zadaje lista svih mogućih vrednosti. Evo kako to kod može izgleda:

```
typedef enum boje  zuta, plava, bele, crvena, zelene };
enum boje pozadine
pozadina    plava
if (pozadina      zelena) {    itd
```

Dakle, definisali smo tip podataka **boje**. Ovaje se vrednosti mogu menjati samo unutar zadate liste. Nakon toga upotrebe promenljivih je sasvim standardna, što pokazuju i dve poslednje linije.

Ako vam se na prvi pogled učini da se ovde radi o nečemu što odskače od C-ovske slobe tipiziranosti podataka, sledeći primer će vas brzo razuveriti:

```
printf("Pozadine je %s\n", pozadina),
```

Ovako nešto će verovatno ispisati „gomilu gluposti" na ekranu – u svakom slučaju neće biti ispisano ono što očekujemo a to je, rečimo reč „plava". Tip **enum** je u svojoj suštini int tip, pa se vrednost promenljive može ispisati u tom obliku:

```
printf( Pozadina je %d\n", pozadine),
```

što će dati 0 ako je pozadina zuta, 1 ako je plava... Int vrednost se dodeljuje svakom simbolu iz liste redom, od nule. Na dodelu vrednosti se može uticati na sledeći način:

```
typedef enum boje  plave, bele  32  crvena, zelena  ,
```

nakon čega će dodela sledećih vrednosti biti nastavljena od mesta promene, pa će **crvena** imati vrednost 33, **zelene** 34 itd.

Retka upotreba **enum**-a u programima je posledica nekoliko činjenica. Prva je slo se ista sivar može postići na daleko uobičajeniji način upotrebom simboličkih konstanti. Sledeći primer funkcionalno je identičan prethodnom:

```
#define zuta 0
#define plave 1
...itd
int pozadina
pozedine    plava  itd
```

Ipak **enum** bi mogao da ima jednu prednost – njegovom upotrebom bi zapravo trebalo da vezemo skup int vrednosti za određenu promenljivu. Ukoliko pokušamo da promenjivoj **pozadine** dodelimo vrednost **sive** komojaler će nas upozoreti na grešku. Ne zalost ovo je tačno samo u lako očiglednom slučaju kada reč **siva** nije definisana nigde u programu ni na koji način.

Nije naime čudno da nam C dozvoljava da napišemo recimo

```
pozadina   16
```

Ako je već **enum** po svojoj prirodi **int** tip, onda je i ovakva „prevara" sasvim u duhu jezika. Ipak, ono što jeste čudno jo je da su simboli iz **enum** liste vidljivi i van deklarisanog tipa. Ako u primeri ubacimo i deklaraciju

```
typedef enum filestatus , opened, closed ,
```

kompajler se uopšte neće pobuniti ako napišemo

```
pozedine   opened
```

Iz ovoga sledi da se u svakoj **enum** listi moraju nalaziti za ceo modul jedinstveni simboli, što ćete lako proveriti eko pokušate sa recimo

```
typedef enum boje_teksta  zuta, plava, crvene  ,
```

Kompajler će odmah upozoriti na redefinisnje konstanti, jer su mu **zuta**, **plava** i **crvene** već poznate iz prethodne deklaracije **boje**.

Ukoliko **enum** lista je, tako je deo C jezika a ne pretprocesore efektivno samo kraći nacin da definišete rastuci niz konstanti, bez potrebe da svakoj pojedinačno dodeljujete ls vrednosi.

#error

Kada sam već kod prelprocesorskih direktiva da pomenem i jednu o kojoj do sada nije bilo reči. Sivan sa uslovnim kompajliranjem se sllom pnlika ponekad iskoristi pikuju i potrebno je razgranalje ispitivanje uslova prelprocesorskim direktivama. Tada se lako može pojaviti i greška u zadavanju parametara. Tipičan slucaj srećemo onda kada okruzenje definišemo sa nikoliko parametara od kojih svaki može imati više različih vrednosti. Umesto da deklarišemo sve moguće kombinacije obično biramo ispitivanje svakog parametra posebno ali se može desiti da određene kombinacije parametara nisu moguće. Tada konsno može posluziti **#error** direktiva kao u sledećem primeru:

```
#if defined( OS2 ) && defined( DIRECTVIOEO )
#error OS2 ne dozvoljava DIRECTVIDEO
#endif
```

#error direktiva tere kompajler da ispiše zedati tekst, da poveca interni brojač grešaka. Nakon toga, proces analize ostatka programe ce biti nastavljen ali program neće biti preveden. Ovo je neće situacija potpuno identična onoj kada kompajler u programu pronađe bilo kakvu grešku.

assert()

Često je, u procesu testiranja programa, daleko jednostavnije ubaciti poneku **printf** naredbu da bismo ispisali trenutno stanje promenjiva nego da dibagerom jurimo grešku. C standardno poseduje i makro **assert** koji na jednostavan način obavlja ovakvu funkciju. Evo jednog primera:

```
#include <assert.h>
#define STRLIMIT 64
char filename[STRLIMIT];
int eppandneme(char *path, char *name)
{
  assert( strlen(path) + strlen(name) < STRLIMIT );
  strcat(path,name);
}
```

Parametar **assert** je bilo kekav logički izraz – ukoliko je njegova vrednost „pogrešno" **assert** ca ispisati

```
Assertion failed:
strlen(path) + strlen(name) < STRLIMIT, file test.c, line 9
```

i program će biti prekinut. U suprotnom, program će normalno nastaviti rad. Posebno je korsno sto **assert** „zna" u kom modulu programe i na kojoj liniji se nalazi, tako da je pronalazenje problematicnog mesta čak i u veoma slozenim programima izuzetno jednostavno. Programi bez iskve bojazni mozete „okititi" neogranicenim brojem **assert** ispitivanja – on će u test fazi biti duzi jer **assert** nosi sa sobom odredeh kod, ali je pre finalnog prevođenja dovoljno da ubacite

```
#define NDEBUG
```

i da sav dijagnostički kod bude potpuno izbačen z programa.

Assert je makro definisan u zaglavlju **assert.h**. Način na koji je realizovan je posebno zanimljiv i jer se konstrukcija koliko nede slicnih osobina C-a i prelprocesora. Deo za glavlja **assert.h** izgleda ovako:

```
#ifdef NDEBUG
#define assert(exp)  ((void)0)
#else
void _FAR _cdecl _assert(void _FAR_  void _FAR
, unsigned)
#define assert(exp) ( (exp) ? (void) 0  _assert(#exp
__FILE__, __LINE__) )
#endif / NDEBUG /
```

Prvo očigledno je da se iza ispisivanja poruke **Assertion failed** pak knje funkcije iz biblioteke **assert**. Njen parametri su **string** kojim je spitan uslov, naz v datoteke, broj programske linije. Makro **assert** u prvom delu deluje sasvim klasično – zamenjuje se ispitivanje izraze koji je parametar makroa i deo **(exp) ?) :** ali zatim sledi niz neobičnosti. Ukoliko je uslov spunjivanja zadovoljen, izvod se deo koda **(void)0**. Ova konstrukcija je „usputn efekat" nastaje kada kompajler prevodi program, nece generisati nikakav kod. Ukoliko uslov ispitivanja nije zadovoljen, poziva se funkcija **_assert** pri cemu joj je prv parametar sam parametar makroa, ali sa prefiksom #.

Ovaj prefiks pripada prelprocesoru i ma zadatak da parametar pretvori u string. To u našem slučaju znači da će se umesto **#exp** funkciji proslediti "strlen(path) < strlen(name) < STRLIMIT" (tekst se propisno ukoviruje znakovima navoda). Ukoliko se iza parametra nađe neki drugi string, biće jednostavno dodat ovome i formiran kao jedan

```
#define test(exp) printf( "izraz je "#exp" \n" )
test(a>3) -> printf( "izraz je a>3 \n")
```

Promenjiva **__FILE__** i **__LINE__** su takode deo pretprocesora i u trenutku obrade programa se zamenjuju stringom – nazivom datoteke – i brojem programske linije.

Sve u svemu, kada se ovako analizira šta radi **assert** makro i odgovarajuca biblioteèka funkcija, očigledno je da je posebnost ove druge samo u tome što je u stanju da prekine program – sve ostalo bi se moglo izvest i bez nje. S obzirom da C poseduje i **abort** funkciju, **_assert** se može lepo zamenit kombinacijom **printf** i **abort**.

Ukoliko nam je potrebno da izbegnamo prekid programa i da program samo ispiše stanje izraza, možemo kreirati sopstveni makro sledeće sadržine

```
#define Info(exp) ( (exp) ? (void) 0  \
  printf(elden  "INFO "\
  #exp  " Uslov nije zadovoljen. file %s, line %d\n", \
  __FILE__, __LINE__) )
```

Možemo izabrati i varijantu u kojoj samo koristimo **__FILE__** i **__LINE__** makroe pa kreirati makro koji će uvek ispisivati izraz i njegovu vrednost

```
#define Infoint(exp) printf(stderr, "INFO file %s, line
%d: "#exp" = %d\n" \
  __FILE__, __LINE__, exp)
Infoint( strlen(filename) );
```

Verujem da ste već naši neki oblik koji će vam odgovarati. Ukoliko odlučite da kreirate svoj makro, obavezno u za glavlje ubacite prethodno i

```
#if defined( NDEBUG )
#define Infoint(exp) (void)0
#else
#define Infoint(exp)
#endif
```

Tako ćete, kao što to i **assert** radi, biti u prilici da definisanjem makroa **NDEBUG** i ponovnim kompajliranjem izbacite ceo dijagnostički kod iz programa.

Nenad Batočanin

Od ovog broja Računara uvodimo novu rubriku. U njoj ćemo se zajedno baviti Clipper-om, njegovim specifičnostima, osobinama i manama. Biće objavljeni izvodi iz interesantnih diskusija na Sezamu, lična iskustva i još mnogo toga. Ako imate neko svoje pitanje na ovu temu, pošaljite ga na redakciju Računara ili preko „Sezama“ kao poruku za konsultanta rubrike.

OD LETA '87 NA OVAMO

Clipper 5.0 je već dosta dugo među nama. Pokazale su se mnoge njegove korisne osobine, ali i neka mane. Nedostaci verzije Summer '87 sada su uglavnom ispravljeni, a može se uočiti i sve jača podrška u vidu raznih biblioteka i razvojnih sistema, što znači da programeri prihvataju novu verziju. Kada se na sve dodaju potencijalna snaga novog Clipper-a, izgleda da se isplati preći na novi Clipper.

Na žalost, stari i novi Clipper nisu potpuno kompatibilni čak ni na nivou sintakse, pa postoje šanse (a Marfi kaže slopocentne!) da vaši stari programi neće proraditi ako ih samo prevedete novim Clipper-om. Obično su potrebna neka ručna doterivanja. Uostalom, u dokumentaciji za verziju 5.01 piše da je u odnosu na verziju 5.0 povećana kompatibilnost sa Summer '87. Nekompatibilnost je nastala iz sasvim razumljivih razloga: autori su bili rešeni između suprotnih koncepcija u staroj verziji i želje da se napravi nešto mnogo moćnije. Rezultat je spao vrlo zadovoljavajući i Clipper će, zahvaljujući tome, veoma teško biti skinut sa vrha popularnosti.

Prva stvar koju ćete uočiti kod prepravke starih programa je različita sintaksa nekih naredbi. Naime, stari Clipper je imao vrlo „labavu“ sintaksu i prevodilac je dozvoljavao svašta u negativnom smislu. Nova sintaksa je znatno strožija, što je bolje, ali zato neki stari programi postaju sintaksno neispravni. Na primer, u verziji Summer '87 savršeno korektna naredba

SUM ALL (Osnova + Dodatak) * Koef TO Suma

u novom Clipper-u neće dobro raditi. Prevođenje će proći bez većih problema, ali će za vreme linkovanja biti javljena greška *Undefined symbol ALL*. To je zato što prevodilac oznaku **ALL** shvata kao naziv funkcije koja ima argument **Osnova + Dodatak**. Još gore je ako ste negde u programu definisali funkciju **ALL**. Tada će program naoko dobro raditi, a u stvari će praviti tešku grešku. Jedini način da se ovo zaobiđe bio je da se promeni sintaksa. Sad ovu naredbu treba pisati na sledeći način:

SUM (Osnova + Dodatak) * Koef TO Suma ALL

U ovom slučaju nema opasnosti od zabune.

MEMORIJSKE KLASE PROMENLJIVIH

Novi Clipper je obogaćen sa dve nove memorijske klase promenljivih: **LOCAL** i **STATIC**. Prostor za promenljive koje se deklarišu kao **LOCAL** ili **STATIC** odvaja se za vreme prevođenja programa, pa je rad sa ovakvim promenljivim daleko brži. Međutim, ako želite da umesto dosadašnjih deklaracija **PRIVATE** i **PUBLIC** koristite **LOCAL** i **STATIC**, vodite računa: jednostavna zamena ne mora da radi u opštem slučaju. Stvar je u vidljivosti (oblasti važenja) promenljivih u raznim modulima. Na primer, nije dobro menjati deklaraciju **PRIVATE** u **LOCAL** bez dodatne provere: su promenljive klase **PRIVATE** vidljive u tekućoj proceduri i u svim koje poziva tekuća, a klase **LOCAL samo** u tekućem modulu, pa vam se u tom slučaju može desiti da dobijete poruku o nedefinisanoj promenljivoj ili još gore, da se kreira nova promenljiva koja sa originalnom nema nikakve veze. Ova osobina promenljivih klase **PRIVATE** mada je u principu protivna pravilima dobrog programiranja, veoma se često koristi. To se vidi u sledećem primeru:

```
PROC Main
  PRIVATE X := 1
  SubRoutine ()
  ? X
RETURN

PROC SubRoutine
  X := 2
RETURN
```

Promenljiva **X** u proceduri **Main** dobiće vrednost 2 i biće vidljiva i u proceduri **SubRoutine**. Međutim, ako se deklariše kao **LOCAL**, biće vidljiva samo u proceduri **Main**, pa će procedura **SubRoutine** automatski kreirati novu promenljivu **X** klase **PRIVATE** koja je sasvim nezavisna od promenljive **X** u proceduri **Main**.

U vezi sa ovim je i korišćenje **PARAMETERS** naredbe. Ova naredba je, kao što znate, korišćena za prihvatanje parametara u procedurama i funkcijama. Tako, ako je bilo potrebno napisati proceduru **Test (X, Y)**, to se standardno radilo na sledeći način:

```
PROC Test
  PARAMETERS X, Y
  ...
RETURN
```

Ovaj, po sasvim subjektivnoj oceni, veoma ružan način pisanja procedura sigurno je smetao svim programerima koji su navikli da parametre pišu u zagradama iza naziva procedure, pa je ta mogućnost u novom Clipper-u dočekana oberučke. Međutim, jednostavna prepravka prethodnog primera u

```
PROC Test (X, Y)
  ...
RETURN
```

potencijalno je veoma opasna! Naime, kada se parametri prihvataju korišćenjem **PARAMETERS** naredbe, kreiraju se promenljive klase **PRIVATE** koje su onda vidljive u toj i u svim procedurama koje se pozivaju iz nje. A ako se parametri prihvate na drugi način, kreiraju se promenljive klase **LOCAL**, pa **nisu** vidljive u procedurama nižeg nivoa.

Sličan je problem i sa makroima: u makroima upotrebljavajte samo promenljive klase **PRIVATE** i **PUBLIC** jer su ostale nevidljive. Na primer:

```
PROC Test
  LOCAL X := 1, Y := 2
  ...
  ? &("X+Y")
RETURN
```

Ako očekujete da ovaj program odštampa 3, prevarili ste se: odštampaće poruku da su promenljive X i Y nedefinisane. Ako deklaraciju **LOCAL** prepravite u **PRIVATE**, sve će raditi kako treba.

IZVRŠAVANJE PROGRAMA, BIBLIOTEKE

Ako ste mali velike programe u kojima ste koristili overleje, vašim mukama je došao kraj! Naime, novi Clipper automatski kreira overleje i upravlja njihovim radom bez ikakve vaše intervencije. Ceo program se deli na fiksni deo i dinamičke overleje koji se učitavaju sa diska po potrebi. **EXE** datoteka će biti znatno veća, ali neka vas to ne brine: pomoću novih sistema za upravljanje memorijom Clipper mnogo bolje koristi memoriju na vašem sistemu. Jedino treba imati u vidu da samo procedure pisane na Clipper-u mogu ući u dinamičke overleje – eksterne procedure pisane na C-u ili asembleru ulaze u rezidentni prostor koji se ne može smanjiti. Iz ovog razloga treba izbegavati statičko alociranje memorije u procedurama koje ste pisali na C-u i asembleru.

Novi Clipper intenzivno koristi svu raspoloživu osnovnu i *expanded* memoriju – ako imate malo EMS memorije, bolje je prepustite Clipper-u nego keš programu. Ako imate dosta memorije, podelite je na keš i Clipper; u svakom slučaju nemojte ostavljati Clipper bez EMS memorije – to će mu drastično smanjiti performanse. Ukoliko pri radu ipak ponestane memorije, Clipper koristi disk i na njega privremeno odlaže sadržaj memorije. To vam omogućuje bezbedno korišćenje veoma velikih nizova i stringova, što je bila velika boljka verzije Summer '87. Ako se računar iz nekog razloga resetuje dok radi program, vrlo je verovatno da će na disku ostati datoteke nerazumljivog naziva, poput **1217543**. To su privremene datoteke koje se kreiraju u slučaju nedostatka memorije. Ukoliko se pojave, možete ih slobodno izbrisati.

Treba reći da novi i stari Clipper nisu kompatibilni na nivou **OBJ** datoteka – nemojte ni pokušavati da ih zajedno linkujete. Znači, sve stare **PRG** datoteke treba ponovo prevesti. Problem su razne komercijalne biblioteke koje ste možda koristili, jer za većinu nije izašla nova verzija koja eventualno podržava novi Clipper. Tu stvari stoje za nijansu bolje, jer su biblioteke obično pisane na C-u i asembleru: novi *extend* sistem za vezu sa ovim jezicima potpuno je kompatibilan sa starim, pa će sve procedure pisane na C-u ili asembleru raditi bez ikakve izmene. To znači, ako je vaša biblioteka „čisto“ pisana (standardne procedure koje koriste *extend* sistem), imate velike šanse da dalje koristite njene mogućnosti. U suprotnom, ne preostaje vam ništa drugo nego da potražite novu verziju biblioteke.

AUTOMATSKA DEKLARACIJA PROCEDURA

Mene je računam ... nervi ralo što u starom Clipper-u ... isti naziv datoteke i procedure u njoj. Kao, ... datoteci **TEST.PRG** pojavila procedura sa nazivom **TEST**, prevodilac je javljao fatalnu grešku *Symbol redefinition error*. To je nasleđe iz dBASE-a gde se u datoteku može upisati jedna procedura, a komunikacija vrši sa **SET PROCEDURE**. Naravno, svako normalan će taj glupi koncept radije menjati za standardni: glavni program je procedura koja se zove **MAIN**, a procedure će se zvati kako god poželite, nezavisno od imena datoteka. Da bi se ova osobina uključila, potrebno je prevoditi programe sa opcijom /n koja sprečava automatsko definisanje procedure istog imena kao datoteka. Ako se ova opcija ne navede, Clipper podrazumeva da se u datoteci nalazi procedura istog imena kao datoteka. Dosta često se desi da se datoteka koja sadrži proceduru **MAIN** greškom prevede bez ove opcije, što proizvodi čudan efekat: kad startuje program, ništa se ne desi, već se samo vrati prompt. Kad se program ponovo prevede, naravno sa opcijom /n i linkuje, sve se postavlja u normalu.

U uskoj vezi sa korišćenjem opcije /n je i korišćenje promenljivih klase **STATIC**. Ova veoma pogodna klasa ima dva oblika korišćenja. Prvi je kad se koristi na početku datoteke, pre svih procedura, tada je promenljiva ove klase vidljiva u svim procedurama unutar datoteke. Ovo je izuzetno korisno, jer omogućava da se kreiraju „globalne“ promenljive za jednu datoteku (što je najčešće i potrebno), bez korišćenja znatno sporijih **PRIVATE** promenljivih koje su uz to nepotrebno i potencijalno opasno vidljive u celom programu. Drugi oblik je kad se deklaracija nađe unutar procedure: onda se kreira promenljiva čija se vrednost ne gubi po izlasku iz procedure, a vidljiva je samo u tekućoj proceduri. Stvar sasvim lepo funkcioniše sve dok se ne zaboravi opcija /n prilikom prevođenja. Na primer:

```
STATIC X := 1

PROC Test
  ? X
RETURN
```

Ako se ova datoteka prevede sa opcijom /n, biće odštampana vrednost promenljive **X** = 1. Ali, ako se prevod izvrši bez opcije /n, podrazumeva se da na početku datoteke stoji deklaracija **PROC** *Naziv datoteke*, pa će promenljiva **X** biti vidljiva **samo** u toj proceduri, što znači da će program skočiti sa porukom o nedefinisanoj promenljivoj **X**.

GET SISTEM

O novom **GET** sistemu je ... u ... broju ... Računara ... rad se o značajno unapređenom sistemu koji ima niz novih opcija za kontrolu polja i vrednosti koje se unose. Nedavno je na Sezamu vođena diskusija u kojoj je konstatovano da je novi **GET** sistem nekompatibilan sa starim, zato što prilikom prekida unosa tasterom **Esc** uništava sve izmene osim na tekućoj **GET** promenljivoj. Na primer:

```
PROC Main
  LOCAL X := 1, Y := 2
  @ 0,0 GET X
  @ 1,0 GET Y
  READ
  ? X, Y
RETURN
```

Kada se ovaj program startuje, mogu se menjati vrednosti promenljivih **X** i **Y** koje inicijalno imaju vrednosti 1 i 2. Ako promenite vrednost promenljive **X** na, primer, na 11 i pritisnete **Enter**, naći ćete se na polju za promenljivu **Y**. Ako sada otkucate 22 i ne pritisnete **Enter**, nego **Esc**, Clipper '87 će obe promenljive vratiti na stare vrednosti 1 i 2, dok će novi prihvatiti promenu promenljive **X**, a samo će **Y** imati staru vrednost.

Da li ste i vi tako mislili? Ako je vaš odgovor **DA**, pod hitno pregledajte slične situacije u vašim programima jer se radi o velikoj zabludi. Naime, jednostavno možete utvrditi da se i stari i novi Clipper ovde ponašaju potpuno identično: **sve** izmene osim poslednje uzimaju se u obzir. Stvar je potpuno ista kao se umesto memorijskih promenljivih uređuju polja iz datoteke. Zato je neophodno, ako želite da omogućite restauraciju starih vrednosti prilikom izmene polja iz tekućeg sloga datoteke, opcija **Quit** ... u ... čuvanje vrednosti, i to u memorijskim promenljivim, a tek po eksplicitnom zahtevu da vrednosti upišete u datoteku.

Od ovog broja „Računara" pokrećemo rubriku u kojoj će biti reči o korišćenju Lotus-a 1 2 3 i Symphony. Nastojaćemo da dodirnemo najrazličitije oblasti rada a poseban naglasak će biti na programiranju u okviru radnih tabela jer na našem tržištu nema odgovarajuće literature. Potrudićemo se da konkretnim primerima olakšamo konstruisanje svakodnevnih radova. Pozivamo i vas, čitaoce koji radite sa radnim tabelama, da dopunjavate raznovrsnost tema. Potrudićemo se da damo najbolje odgovore i savete. Pitanja, predloge i sugestije možete poslati poštom na adresu redakcije ili preko Sezama

SNIMANJE DATOTEKE

Jedna od najčešće korišćenih makro procedura je ona koja se odnosi na snimanje datoteka na disk/disketu. Da vas podsetim iz više razloga je neophodno da vam prede u naviku da povremeno snimate tabelu sa kojom radite nestanak električne energije uključivanje nove tabele a niste snimili onu na kojoj ste do tada radili itd. Postupak za snimanje tabele na disk/disketu u *Lotus-u 1-2-3* je sledeći: pozovete meni (/) komande **File** pa **Save** upišete ime datoteke i pritisnete ENTER (↵). U svakom sledećem snimanju morate još aktivirati i komandu **Replace** kojom „dopunjujete" sadržaj datoteke novim podacima. Makro procedura koja će odraditi isti posao samo brže bi izgledala kao na **slici 1**. Bilo je da upis procedure počnete nekim od predznaka teksta (' " ^ \).

U *Lotus Symphony* komande su nešto drugačije: pozovete SERVICES (F9) meni aktivirate komande **File Save** upišete ime datoteke i pritisnete ENTER (↵). U svakom sledećem snimanju morate još aktivirati i komandu **Yes** kojom „dopunjujete" sadržaj datoteke novim podacima. Makro procedura koja će odraditi isti posao, samo brže izgleda kao na slici 2.

Da bi makro procedura radila namestile pointer na naziv makroa (**\ a**) i izdajte komande **/Range Name Labels Right** ENTER (u *Lotus-u 1-2-3*) odnosno MENU (F10) **Range Name Labels Right** ENTER (u *Symphony*). Pokretanje procedure se vrši tako što se pritisne taster Alt slovo S. Vaša tabela će biti snimljena na disk.

	A	B	C
1	**Naziv**	**Makro**	
2	**makroa**	**procdura**	
3	\s	/fsPRIMER~r	
4			
5		*Slika 1*	

Makro procedura za ubrzani Save (Lotus)

	A	B	C
1	**Naziv**	**Makro**	
2	**makroa**	**procdura**	
3	\s	{S}fsPRIMER~y	
4			
5		*Slika 2*	

Makro procedura za ubrzani Save (Symphony)

RAČUNANJE VREMENA

Vrlo često se javlja potreba da u radnoj tabeli na određenom mestu imamo tekući datum i vreme. Obično se oni postavljaju uz izveštaje koje treba svakodnevno štampati pa u skladu s tim treba da se zna na koji vremenski period (datum i vreme) se odnose. To praktično znači da ako hoćemo u tabeli da prikazemo tekući datum i vreme u obliku

21 Dec-91 13 16

moramo angažovati dve susedne ćelije sa funkcija ma @NOW. Na jednu ćemo zadati **/Range Format Date 1** komandu... a na drugu **/Range Format Date Time 4** (odnosi se na *1 2 3* Ver 2.2). Prilično jednostavno. Što je u ostalom i odlika *Lotus-a 1 2 3*. Šta međutim ako se zainteresujemo da hocemo da iskoristimo samo jednu ćeliju i da pomocu jedne funkcije odnosno izraze dobijemo isti rezultat? U tom slučaju nudim vam jedan primer dovoljan za razmišljanje na teme o delumima vremenu i programiranju

Funkcije	Rezultat (broj)	Funkcije (string)	Rezultat string)
@DAY(@NOW)	22	@STRING(@DAY(@NOW),0)	22
@MONTH(@NOW)	2	@CHOOSE(@MONTH(@NOW),0,"Jan","Feb","Mar"..	Feb
@YEAR(@NOW)	91	@STRING(@EAR(@NOW),0)	91
@HDUR(@NDW)	10	@STRING(@HOUR(@NOW),0)	10
@MINUTE(@NOW)	38	@RIGHT("0"&@STRING(@MINUTE(@NOW),0),2)	38

U jednu ćeliju treba upisati sledeću funkci ju

```
STRING( DAY( NOW),0)&"r"& CHOOSE( MONTH( NOW),0,"Jan","Feb"
,"Mar","Apr","Maj","Jun","Jul","Avg","Sep","Okt","Nov","Dec")&"i"
& STRING( YEAR( NOW),0)& rr & STRING( NOUR( NOW),0)&" " &RIGHT("0"& STRING( MINUTE( NOW),0),2)
```

NAPOMENA. Upis unošenje izraza izvršite u nepreklopnutom nizu u jednu ćeliju. Ovde je funkcija prikazana u nekoliko redova jer je dugačka. Dobro je pažnju na deo „r". Na tom mestu treba uneti razmak (blankove) a slavljeno je slovo i radi lakšeg orijentisanja. Znači ne treba kucati i nego razmak. Ova funkcije radi u *Lotus-u 1 2 3* Ver. *2.x* i *3.x* kao i u *Symphony* Ver. *2.x*.

Da pojasnim malo o čemu se radi. Ovde se polazi od ideje da se tekući datum i vreme prikažu kao string za razliku od ubičajenog numeričkog. Podsetiću vas funkcije @NOW @DATE i druge kao rezultat vraćaju broju vrednost, koju pomocu **IRange Format Date (Time)** komande prikazujemo u nama najboljem obliku. Dakle funkciju unosimo u jednu ćeliju a vrlo nam se string koji „pokriva" dve susedne ćelije.

Funkcija @STRING(x n) pretvara vrednost **x** u string sa **n** decimalnih mesta. Pri tome **x** može biti bilo koja vrednost a **n** ceo broj od 0 do 15. Znači nas zadalak je da pomocu odgovarajućih @ funkcija nademo vrednosti (sećska brojeve) za tekući dan mesec godinu sat i minut pa da ih konvertujemo i spojimo u jedan string i prikazemo na odgovaraluci način. Veza izmedu stringova se ostvaruje karakterom „&"(ampersand). Pomenule senjske brojeve dobicemo pomoću funkcija @DAY @MONTH @YEAR @HDUR i @MINUTE kod kojih je argument funkcija @NDW (vidi „Recunar" 69/52). Čini mi se da će se razlika između rezultata koje daju @ funkcije i @ string funkcije najbolje videti iz priložene tabele radene u februaru 1991. godina.

Ako obratite pažnju videcete da su brojne vrednosti poslrojene sa desne strane ćelije a stringovi sa leve. Dakle vidljivi su rezultati string funkcija.

Funkcija @MDNTH(@NDW) vraca broj meseca u godini (februar=2). Da bismo mesec prikazali kao string iskoristićemo funkciju @CHDDSE(offset list). Dna nalazi vrednost ili string u listi specificiran od **offset**. U ovom slučaju **offset** @MONTH(@NDW) vraca broj 2 sa 0 decimalnih mesta. @CHOOSE na osnovu tog rezultata iz liste vadi string „Feb".

Da bismo obezbedili da se string za minute sastoji uvek od dva znaka mozemo iskonstiti funkciju @RIGHT(string n). Dne vraća zadnjih n karaktere u stringu. U nasem slucaju ispred je dodeta nula („0") leko da se obezbedi dvocifren" string za minute (na primer 04 min.). Ukoliko minuti imaju stvarno dve cifre dodata nula se nece videti jer je u nasoj funkciji n=2.

Dakle ako ste sve ovo uradili i nadam se razumeli sada delove treba sastaviti u celinu. Kako vidi se na početku u kompletnom izrazu. Znaci vezu između stringova ostvarićete pomocu ampersand karaktera (&). Pojedine delove mozete razdvojiti proizvoljnim brojem razmaka (znak i u izrazu) zalvorenih u znake navoda („).

Ovaj način prikazivanja tekuceg datuma i vremena mozete koristiti u raznim situacijama. Da li je bolji ili zgodniji od ubičajenog prosudićete sami na kon prve upotrebe.

OBAVEZNA LEKTIRA

Inside Lotus 1 2-3 Macros (Izdavac Brady New York USA strana 615 cena $48.95) jedna je od knjiga koju bi svaki ozbiljniji konsnik *Lotus-a 1 2-3* pa cak i *Symphony* trebalo da ima. Nije jeftina ali vredi svaka ulozene pare. Podnaslov knjige je „The Definitive Guide and Reference" i to je apsolutno tako. Čini mi se da nije moguce na bolji nacin obraditi i objasniti makro procedure i programiranje u *1 2-3* nego što je to učinjeno u ovoj knjizi. Mozda to ne bi trebalo de nas cudi jer autori su istaknuti članovi Lotus Development. Jedan je (Richington) urednik u časopisu „Lotus Magazine" a drugi (Tucker) direktor proizvodnje u odeljenju za PC tadne labele. Dakle, ljudi koji maksimalno vole i poznaju ono o čemu pišu.

Autori nes postepeno uvode u vode programiranja u *1 2-3*. U prvom poglavlju su objašnjenja o makroima zašto se koriste kako se knjiga koristi slično uvodne teme. Dalje kroz ostala poglavlja slede objašnjenja do najsitnijih detalja o tome kako se prave jednostavne makro procedure pa sve složenije. Je do onih koje spadaju u pravu ezoteriju. Knjiga je obimna jer autori koriste masu primera u kojima navode šta i gde treba nešto upisati da bi neka makro procedura funkcionisala.

Mozda je najinteresantniji deo u kome se do detalja opisuju tehnike programiranja. Sve makro komande su podeljene u nekoliko poglavlja: kontrola ekrana, interaktivni rad, kontrola toka programa, manipulacija podacima, manipulacije datotekama, pristup Add-In programima. Za svaku komandu je dato detaljno objašnjenje i bar jedan primer. Ako ne više.

U nastavku je dat pregled @ funkcija koje se koriste u makroe. Po istom sistemu: uz detaljna objašnjenja ide i obilje primera.

U poslednjem odeljku dale su neke napredne tehnike programiranja. Tu mozete naucili mnogo o nacinima za unos podataka o različitim mogucnostima kretanja sopstvenih menija, upotrebi ekrana HELP sistemima strukturnom programiranju. Za nimljivi su delovi o testiranju aplikacije i otkrivanju grešaka. Ne o necinima da se one izbegnu.

Interesantno je i poglavlje u kome autori iznose bazni koncepl o pristupu pri izradi aplikacije. Tej deo je univerzalan i njega bi trebalo da pročitaju svi koji se bave programiranjem odnosno pravljenjem aplikacija za druge korisnike.

Dakle rec je ne samo po mom mišljenju o jednoj izuzelnoj knjizi koju bi svakako trebalo imati u biblioteci. Naravno ako radite sa *Lotus-om 1 2 3* ili *Symphony*. Izuzetno korisna stvar je priložena disketa na kojoj su svi primeri iz knjige tako da su vezba i pracenje stiva znatno olakšani. Datoteke su nazvane prema oznakama slika u knjizi. Ako niste početnik preskočićete poznate stvari i skoncentrisaćete se na poglavlje – datoteke koja vas interesuju. Mislim da nema korisnika *Lotus-a 1 2 3* koji ne bi imao šta da nauci iz ove knjige.

441 [COMM.HAYES, PC] Od dana kada je Sezam omogućio čuvanje neprenesenog PAD a za sledeći poziv ponekad se dogodi da telefon zvoni u prazno dok pozvani node arhivira PAD prethodnog korisnika U tom slučaju treba pustiti modem da zvoni i sačekati desetak sekundi da se operacija završi posle čega će se veza normalno uspostaviti Međutim ma koliko ste pritiskajući PLUS ili neki taster karakterističan za vaš komunikacioni program produžili ovo čekanje modem će posle 30 tak sekundi „spustiti slušalicu" i ispisati NO CARRIER Ne ostaje ništa drugo nego da se ponovo zove

Uzrok je u registru S7 – čak i oni koji čitaju uputstvo za modem lako pravide tu „silnicu" Za taj registar obično piše *Wait time for carrier after answer Factory default 30* To znači da će modem posto završi sa biranjem broja čekati na vezu maksimalno 30 sekundi a onda zalupiti slušalicu bez obzira sta vi o tome rekli komunikacionom programu! Kada znamo u čemu je problem do rešanja se stiže jako brzo – u inicijalizacionu sekvencu modema treba ugraditi S7 255 sto vam daje preko 4 minuta za uspostavljanje veze Na žalost S7 se ne može „zapamtiti" u EEPROMU već se sa svakim resetom modema postavlja na 30 pa inicijalizacija jednim elegantnim ATZ više ne prolazi [Preuzeto sa Sezama prilog **Aleksandar Vasić** (*vasic*)]

442 [BIOS DTK, PC] Neprijatan bag (ili bar nekompatibilnost) DTK BIOS a Kao što smo već pisali u ovoj rubrici prilikom „podizanja" sistema BIOS proverava vrednost na adresi 40h 72h i ako tu nađe vrednost 1234h izvršava tzv *warm boot* tj ne testira hardver i memoriju DTK BIOS takođe proverava tu adresu ali ne traži 1234h nego 0A55Ah (Zašto? Zaista nije jasno mada je broj A55A kada ga binarno napišete dosta „zanimljiviji" od 1234h) Zbog ove promene kombinacija Ctrl Alt Del u prisustvu DOS ovog programa KEYB ne radi kao *warm* nego kao *cold boot* (da bi proradila treba naći niz bajtova 34h 12h u KEYB COM i zameniti ih sa 5Ah 0A5h) Takođe ni 4DOS naredba REBOOT ne može da izvede *warm boot* da bi se to ispravilo treba naći 0BAh 34h 12h u 4DOS COM i zameniti sa 0BAh 5Ah A5h Moguće je da se slični problemi jave i u drugim programima [Preuzeto sa Sezama prilog **Nenad Bogojević** (*nkbog*)]

443 [TEHP, PC] Jedan od noviteta DOS a 5 0 (a ranije i DOS a 4 XX) je serijski broj diska koji se može koristiti i kao vid zaštite programa Treba naravno napisati funkciju koja će taj broj pročitati u čemu može da vam pomogne opis sa slike 1 [Preuzeto sa Sezama prilog **Predrag Todorović** (*todorp*)]

444 [MIKR, PC] Mora se priznati da su IBM i Microsoft uneli priličnu zbrku među korisnike sa imenima pojedinih zona memorija Osnovna memorija od početka adresne mape do 640K UMB (*Upper Memory Block*) između 640K i 1024K i najzad HMA (*High Memory Area*) na adresama 1024K – 1088K koja jeste iznad granice prvog megabajta ali joj se na 286 i 386 može pristupiti kao da je ispod te granice

Dok su prva dva termina uglavnom jasna HMA je tajanstvena – ili je ispod granice ili nije zar ne? U stvari 80286 i 80386 sve vreme generišu adresu sa više od 20 bitova a u realnom režimu važi relacija

efektivna adresa – 16 seg+offset

pa ako seg uzme maksimalnu vrednost (0ffffH) najveća adresa koja može da se dobije sa 16-bitnim ofsetom je 1088K-1 dakle 64K više od megabajta Na AT u su u normalnom režimu rada hardverski isključene adresne linije preko A19 potrebne za adresiranje više od 1M pa se adrese preko 1M „premotavaju" na početak adresnog prostora (u prvih 64K) Ako se te adresne linije aktiviraju (ne samo A20 nego i one iznad nje) moguć je direktan pristup tom dodatnom bloku (UMB) od 64K iznad prvog megabajta [Preuzeto sa Sezama prilog **Aleksandar Damnjanović** (*alexa*)]

445 [BASE CLIPPER, PC] Kako da saznate da li neki hard disk ili particija postoji ili ne? Najjednostavniji način je funkcija DISKSPACE(N) gde je N oznaka diska – 0 za A 1 za B 4 za D i tako dalje Ukoliko DISKSPACE vrati nenormalno veliki broj otprilike 1152 terabajta particija ne postoji Ideja očigledno važi samo ako u sistemu nema particija većih od 1152 Tb! [Preuzeto sa Sezama autor **Jovan Bulajić** (*bulaja*)]

446 [BASE CLIPPER PC] *Clipper* za indeksiranje koristi modifikovano B-stablo (BTREE) koje se sastoji od zaglavlja (prva strana) i potrebnih pointera odnosno ključeva (ostale strane) Kompletna struktura data je na slici 2 [Preuzeto sa Sezama prilog **Zoran Kehler** (*zkehler*) prema knjizi Rick Spence Clipper Programming Guide" 2nd Edition]

447 [HARD PC] Da li ste primetili da po instalaciji DOS-a 5 0 i stavljanja DOS=HIGH u CONFIG SYS vaš AT postaje gotovo 30% sporiji? To se dogodilo nekolicini korisnika Sezama na kraju se pokazalo da je (iz neispitanih razloga) „krivac" mikroprekidač na matičnoj ploči koji se zove ROM Speed" Simptom ovog problema je da u Turbo modu DOS =HIGH usporava računar dok na nižoj frekvenciji DOS=HIGH ne izaziva nikakve promene Rešenje je premeštanje džampera JP9 (ROM SPEED CONTROL JUMPER) sa pozicije 1 2 (gledano od prednje strane kompjutera) u položaj 2-3

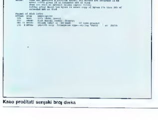
Ovo se odnosi na SunTac 286 12(16) matičnu ploču ali možda pomogne i ostalim korisnicima AT a [Preuzeto sa Sezama prilog **Svetozar Nedeljković** (*sveta*)]

448 [MREZ NOVELL, PC] *Novell 3 XX* zahteva postojanje male (po potrebi i veće) DOS particije dok se na ranijim verzijama (npr 2 15) ceo disk mora posvetiti *Novell*-u Bilo bi međutim zgodno da postoji mala DOS particija ako ništa drugo da bi se lakše pokretao VREPAIR ili neki drugi uslužni program O mrežama koje koriste *non-dedicated* server da i ne govorimo

Glavni problem je u tome što *Novell* zahteva da *Netware* particija bude na početku diska a DOS ova particija opet mora počinjati u okviru prvih 32 MB diska što praktično znači da se (poštujući ova ograničenja) mogu iskoristiti samo diskovi do 40MB koji su premali da bi zaslužili mesto u serveru neke mreže

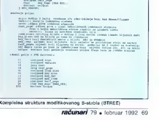
Rešenje se sastoji u tome da se COMPSURF-om ili NETWORK DM om definiše *Netware* particija koja je za 1-2MB (prema potrebi) manja od celog diska a ostatak kao 16 bitna R/W particija *Netware* particiju treba označiti kao nebutabilnu a 16 bitnu kao butabilnu Takva particija (16 bitna) se kasnije tor

Slika 1



Kako pročitati serijski broj diska

Slika 2



Kompletna struktura modifikovanog B-stabla (BTREE)

matira običnom DOS komandom FORMAT /S. Na tako formatiran DOS disk se snimi sadržaj dotadašnje STARTUP diskete (NET$OS i ostalo) i sistem je spreman za butovanje sa diska. *Novell* se može pokrenuti iz AUTOEXEC-a ili nekom .BAT procedurom. [Preuzeto sa Sezama, prilog **Živorad Marković** (*makas*)]

449 [COMM, VAX] Svima koji su slali elektronsku poštu preko VAX-a poznati su problemi sa datotekama koje VMS pre slanja ili pri prijemu „unakazi" ubacujući dodatne oznake kraja sloga na ono što mu liči na liniju dužu od 256 znakova – docnije se mora koristiti neki program da se takvi kodovi eliminišu. Na sreću, na VMS-u 5.4 postoji (nedokumentovana!) opcija za slanje datoteka uz čuvanje strukture – treba samo otkucati (po startovanju programa MAIL, naravno) SEND ime.ext /FOREIGN a zatim navesti adresu, temu i ostale stvari na uobičajeni način. Opcija /FOREIGN se inače često koristi pri radu sa magnetnim trakama, a sada je našla svoje mesto i u MAIL-u ... nejasno je samo zbog čega nije dokumentovana. [Preuzeto sa Sezama, prilog **Radivoje Zonjić** (*zonjic*)]

450 [OS VMS, VAX] Verzije VMS-a iznad 4.0 omogućavaju sistem administratoru da ograniči trajanje lozinke nekog (ili svakog) korisnika – posle određenog intervala (npr. 3 ili 6 meseci) sistem zahteva od korisnika da promeni lozinku i ne pušta ga da nastavi rad dok to ne učini. Ne znamo koliko je ovo korisno sa bezbednosnog aspekta, ali znamo da se do skora dalo lako zaobići – na zahtev sistema promenite lozinku u 123456 a onda sa SET PASSWORD vratite staru lozinku i ... mirni ste sledećih pola godine!

Još novije verzije VMS-a ovo ne omogućavaju – sistem pamti nekoliko poslednjih lozinki i, ako pokušate da vratite neku od njih, ispisuje poruku „Password found in history list". Sve ovo pišemo da bismo vas upozorili da ne pokušavate da budete „pametniji" od VAX-a i da nekoliko puta menjate lozinku nadajući se da ćete „prepuniti" ovu listu i biti u prilici da ponovo izaberete svoju staru lozinku. Posle 50-tak promena lista će se zaista popuniti, ali posle toga nećete **uopšte** moći da menjate svoju lozinku, nego ćete morati da koristite SET PASSWORD /GENERATE i prihvatite neku od „levih" lozinki koje vam sam VAX nudi.

Ukoliko vam se ovo već desilo, nema druge nego da se obratite nekom od sistem programera i zamolite ga da vam obriše listu starih lozinki. [Preuzeto sa Sezama]

451 [IGRE PRINCE 409, PC] Startovanjem programa *Prince of Persia* komandom PRINCE MEGAHIT (umesto „normalnog" PRINCE) ulazite u mod koji je očito namenjen testiranju (ili demonstriranju) programa. Na raspolaganju su vam tada nove komande: **Shift+W** (omogućava skok sa bilo koje visine, dok muzika traje), **Shift+T** (dodavanje života), **Shift+I** (okretanje celavog ekrana), **Shift+L** (prelazak na sledeći nivo), **Shift+B** (gašenje svetla), **H** (pregled levog susednog ekrana), **J** (pregled desnog susednog ekrana), **U** (pregled gornjeg susednog ekrana), **N** (pregled donjeg susednog ekrana), **K** (ubijanje protivnika na ekranu), plus (dodavanje vremena) i minus (oduzimanje vremena). [Preuzeto sa Sezama, prilozi **Boško Radivojević** (*bole*) i **Saša Mitrović** (*smitrovic*)]

Priloge za ovu rubriku šaljite na adresu „Računari" (za „Bajtove lične prirode"), Bulevar vojvode Mišića 17, Beograd ili preko Sezama u vidu lične poruke uredniku (*mail write dejanr*)

WORD PERFECT FOR WINDOWS

Nastavak sa strane 37

Očekivalo bi se da više prostora posvetimo *WordPerfect*-ovom *File Manager*-u, ali nismo stekli utisak da se radi o nekom posebno značajnom novitetu. *WordPerfect* je oduvek imao neku vrstu sopstvenog PCTools-a koji se startovao pritiskom na *List* (F5) i obezbeđivao pronalaženje, kopiranje, brisanje i druge operacije sa datotekama potrebne tokom pisanja. Slična logika je uvedena i u *Windows* verziju, pri čemu je *File Manager* nešto pogodniji od *Windows*-ovog utoliko što omogućava prolazak kroz više nivoa kataloga. Na ekranu se prikazuje sadržaj direktorijuma iz koga smo pošli, zatim direktorijum u koji smo sledeći prešli itd, pri čemu su svi ti direktorijumi vidljivi u isto vreme, tako da možemo aktivirati datoteke iz bilo kog od njih. Lepa ideja, ali ne posebno korisna. Daleko više nam se dopala *Quick list* opcija u **svim** menijima paketa *WordPerfect for Windows* – ako stalno radite sa datotekama u nekoliko direktorijuma, tim direktorijumima možete da dodelite logička imena (npr. pisma, tekstovi, slike itd) i da ih onda pod tim imenom pronalazite u spisku, umesto da se „šetate" kroz čitavo stablo diskova i kataloga.

Pomenimo još i povećani broj dokumenata koji se mogu otvoriti (*WordPerfect 5.1* je omogućavao istovremeni rad sa svega dva dokumenta (što je obično bilo dovoljno, ali je ponekad treći dokument znao da zatreba) a *WordPerfect for Windows* omogućava otvaranje njih 9), poboljšanu opciju za upoređenje dokumenata na kojima je radilo više ljudi, mogućnost definisanja fonta koji će važiti u svim fusnotama, proširenja opcija GOTO, mogućnost učitavanja dokumenata u raznim formatima bez potrebe za programom CONVERT, snimanje jednačina u posebne .EQN datoteke koje se onda mogu prenositi iz teksta u tekst, manje izmene u programu za kontrolu spelovanja... Naravno, i HELP je znatno proširen, a rad sa njim žestoko olakšan, ali nam se čini da je to pre zasluga mehanizama samog *Windows*-a nego programa *WordPerfect*.

Dokumenti snimljeni pomoću *WordPerfect*-a 5.1 i 5.0 (štaviše i pomoću *WordPerfect*-a 4.2) se nepogrešivo učitavaju u *WordPerfect for Windows* bez ikakve potrebe za nekom specijalnom komandom ili konverzijom. Moguće je, naravno, da se neke stvari u učitanom dokumentu promene (malo šira slova, margine itd) ali će sve sa nekoliko poteza pera (ovaj, miša) biti dovedeno u red. *WordPerfect for Windows*, da bi kompatibilnost bila potpuna, može da snima na disk kako u svom formatu tako i u formatu prethodnih verzija 5.0 i 4.2, kao i u brojnim drugim formatima kao što su *Ami Pro*, ANSI, *Display Write*, *MS Word*, *MS Word for Windows*, *Multi Mate* itd. Ne postoji nikakva razlika po pitanju formata dokumenata između DOS i *Windows* verzije *WordPerfect*-a 5.1 – možete slobodno razmenjivati dokumente između ova dva paketa.

Dokumentacija je i dalje na uobičajenom nivou – opširan referativni priručnik koji se za ovu priliku smanjio na 850 strana (*WordPerfect 5.1* je imao 1100 strana uputstva, da li je to zato što je jednostavnije raditi sa *Windows*-om ili zato što se više ne pominje rad na sistemu bez hard diska koji je odnosio silan prostor, nismo pokušavali da utvrdimo). Naredbe su u njemu naslagane abecednim redom, što nije baš pogodno za učenje ali je izuzetno zgodno kada vam zatreba detaljniji opis neke opcije. Početnicima je namenjen *WorkBook* i, naravno, *Learning* disketa čiji se sadržaj u zadnjih par godina nije previše menjao. Sve to je praćeno izvesnim brojem READ.ME i .DOC datoteka u kojima su pobrojane stvari koje iz nekih razloga nisu stale u uputstvo. Primetili smo i neke sitne izmene proizašle iz neprijatnih iskustava korporacije *WordPerfect* – dok je, na primer, u prethodnim verzijama uputstva pisalo da su dokumenti zaključani lozinkom zauvek nepristupačni, sada se pominju „eksperti kriptoanalitičari koji mogu da otkriju sadržaj originalnog dokumenta samo na osnovu šifrata".

WORDPERFECT I STRANI JEZICI

Od *WordPerfect*-a *for Windows* smo sa pravom očekivali da konačno reši problem stranih slova – još od verzije 5.0 na raspolaganju je detaljan set od preko 5000 znakova koji obuhvataju naše latinična slova, kompletnu ćirilicu (kako rusku, tako i srpsku, bugarsku, makedonsku itd), ali su se neka od tih slova (npr. kompletna ćirilica) na ekranu pojavljivala kao kockice. *Windows* verzija zaista obezbeđuje prikazivanje **svih** slova na ekranu što znači da je *WordPerfect for Windows* savršeno opremljen za rad u našim uslovima – uz mali trud da predefinišete tastaturu i dodelite znake odgovarajućim tasterima, imate tekst procesor spreman za ćirilicu i latinicu? Na žalost, ova lepa vest ima i svoju mračnu stranu – ovako generisana slova na papiru izgledaju loše, a na ekranu još gore: očito se konstruišu na osnovu nekih vektorskih zadatih fontova koji dosta odudaraju od ostalih veoma dopadljivo dizajniranih bit-mapiranih slova.

Problem sa štampačem se rešava korišćenjem latiničnih i ćiriličnih fontova (prekopirali smo drajver od DOS verzije i bez problema „progovorili" ćirilicu), ali sa ekranom nije lako – format u kome *WordPerfect for Windows* „čuva" znake nije dokumentovan, što znači da „pečovanje" nije baš jednostavan posao. Istini za volju, u okviru dokumentacije se pominje da postoje dve verzije datoteke WP.DRS u kojoj se čuvaju strana slova: jedna verzija je namenjena američkom tržištu i zauzima manje memorije uz lošiji izgled stranih slova, a druga je namenjena izvozu. Nismo znali kako da proverimo **koja** je verzija isporučena uz naš *WordPerfect for Windows* (baš su mogli da daju i obe!) ali se iskreno nadamo da je to ona lošija i da se promenom može doći do lepših slova; uostalom, ako se ispostavi da su **ovo** ta lepša slova, zbilja ne možemo da zamislimo kako bi izgledala ona ružnija!

Očito je uložen veliki trud u podršku stranih jezika – hifenacion modul, rečnik, rečnik sinonima i mnoge druge stvari sada zavise od izabranog jezika, a najavljeni su i paketi za raznorazne evropske i istočnjačke jezike među kojima se nalazi i hrvatski (srpski, na žalost, još ne). Da li to znači da ćemo jednog dana dočekati da nam korporacija *WordPerfect* napravi ono što godinama želimo, ali se nikada nismo usudili da uradimo – program za kontrolu pravopisa prilagođen našem jeziku, ostaje da se vidi. Vredi pomenuti da su sve poruke koje program automatski ugrađuje u tekst sada izdvojene u datoteku WP.LRS (LRS – *Language Resource*) čija je struktura dokumentovana – nema, na primer, nikakve prepreke da umesto *(Continued)* umetnemo odgovarajuću srpsku reč i tako „sredimo" fusnote. Podrazumeva se, naravno, da datoteku WP.LRS treba vrlo pažljivo menjati, pošto je *WordPerfect for Windows* pri učitavanju sintaksno ne kontroliše – neispravna datoteka može izazvati neispravan rad programa.

JOŠ KAD BI HTEO DA RADI...

Ostaje još da odgovorimo na pitanje kako sve te pote koje smo opisali opisali rade u praksi. Ukratko rečeno, jako loše. Toliko loše da je ova verzija *WordPerfect*-a *for Windows* neupotrebljiva za bilo kakav ozbiljan rad – čak i kada radite sasvim obične stvari vrlo su česte (među korisnicima *Windows*-a ozloglašene) UAE poruke (*Unrecoverable Application Error*), zatim direktna izletanja u DOS (!) i razni drugi oblici zaglupljivanja koji se manifestuju na milion načina, ali imaju jedan zajednički faktor – šta god vam se desi, ostajete bez teksta u memoriji! Ukoliko dakle pokušate da radite sa programom, ne zaboravite da ga naterate da automatski snima tekst posle svakog minuta rada.

Napominjemo da program koji smo testirali nije nikakva beta verzija nego komercijalno izdanje i da smo ga isprobali na nekoliko računara u raznim konfiguracijama. Uvek isto – malo rade, pa gubitak teksta. Slične probleme prijavljuju i američki korisnici na BIX-u, čudeći se kako jedna korporacija tako visokog renomea može sebi da dopusti da izbaci na tržište **toliko** bagova i neupotrebljiv proizvod!

Sa dobre strane, *WordPerfect for Windows* je izuzetno brz. Nećemo, naravno, da kažemo da u nekim trenucima ne bi mogao da bude i brži (naročito pri startovanju), ali je u poređenju sa *Microsoft*-ovim programom *Word for Windows 1.1* (verziju 2.0 još nismo videli) prava munja. Moramo zapravo reći da nismo očekivali da vidimo *Windows* aplikaciju ovako dobrih performansi!

Šta reći na kraju? Kada bismo verovali da će *WordPerfect for Windows* ostati u ovom stanju, ovaj prikaz ne bi imao smisla jer vam program takav kakav je ne možemo preporučiti ni za šta – osim za gledanje. Pa ipak, verujemo da će *WordPerfect* veoma skoro (možda već u danima kada budete čitali ovaj tekst) izbaciti sledeću (već najavljenu) verziju paketa *WordPerfect for Windows* koja će imati manje bagova i čije će performanse, kvalitet ispisa i, naročito, superiorni korisnički interfejs ukrašen sa par fundamentalnih noviteta predstavljati dobru osnovu za najbolji tekst procesor 1990-tih što, uostalom, i očekujemo od *WordPerfect*-a. Ostaje samo da sačekamo i vidimo.

SezamFile

Priprema: Jovan Bulajić **4**

Sa više od 4 MB novih datoteka u Sezamovim PC direktorijumima, decembar je do sada rekordan mesec. Uz redovni mesečni izveštaj, ovog puta dajemo i malu godišnju statistiku, kao i opis jednog vrlo korisnog programa.

NOVE DATOTEKE

Među arhiverima vlada umereno zatišje – pred buru koja nas, verujemo, uskoro očekuje. Izašla je verzija 2.22 ARJ arhivera, uz minorne izmene u odnosu na prethodnu, ali se zalo priča o beta verziji 2.3, koja konačno donosi i poboljšanja u performansama u odnosu na 2.0. Novi PkZIP je još uvek u beta verziji (1.93a), i nadamo se skoroj pojavi *ZIP 2.0*. S obzirom da su arhiveri postali neophodna alatka u svakodnevnom poslu, pretpostavljamo da većina korisnika *Sezama* sa nestrpljenjem iščekuje dalje događaje.

Na polju komunikacionih programa dešavalo se dosta toga. Izašla je još jedna beta verzija (izgleda da su one postale moda u zadnje vreme) *Telemate 2.17*, a očekuje se i skora (dugo očekivana) pojava prave verzije 3.0. Pažnju je izazvao mali komunikacioni program iz časopisa „Byte", koji možda po osobinama nije ništa posebno (ali poseduje sve što je neophodno), ali je zato kompletan dat u izvornom kodu (C). Stigle su i nove verzije u ovoj rubrici već pominjanih programa *DSZ & GSZ* (Zmodem) i *Tele-Replica*.

U WINDOWS direktorijum su takođe dospela dva nova komunikaciona programa. Jedan od njih, *Unicom 3.0*, verovatno je do sada najbolji komunikacioni program za prozore, ali je neregistrovana „shareware" verzija (kakva se nalazi na *Sezamu*), na žalost, prilično nepogodna za rad (zbog stalnog ispisivanja upozoravajućih poziva na registraciju).

Programeri su, uz već spomenuti komunikacioni program u izvornom kodu, najviše bili obradovani novom verzijom liste (tekstualne baze podataka) svih mogućih (i nemogućih) PC interapta, a za ljubitelje starina pristigao je Cobol kompajler.

Krajem meseca (pred same novogodišnje praznike) korisnike *Sezama* naročito su zainteresovale dve nove igre – *Commander Keen 1* i *Duke Nukem*. Obe su tipa arkadnih avantura (nismo baš neki stručnjaci za ovu oblast softvera, ali valjda se to tako zove) i uz dobru ideju i još bolju realizaciju (EGA/VGA grafika, gomile spratova i čudovišta koja vas proganjaju) pružile su obilje zabave u danima odmora. Nadamo se da su oni koji imaju svoju potrebu za pucanjem (pa makar i samo na male i velike zelene) zadovoljni za instalaturom.

Ukratko i o preostalim novitetima: nove verzije *TSR Utilities* i *VDE* tekst editora, generator fraktala (sjajna stvar za isprobavanje mogućnosti vaše nove VGA kartice), simbolički matematički kalkulator i još dosta uslužnih programa.

1991. POD LUPOM

Jedna godina je završena, pa je vreme i za malo statistike. Ovog puta ćemo to uraditi samo za devet meseci, pošto je detaljnija statistika za datoteke na *Sezamu* počela da se vodi tek negde od aprila prošle godine.

U tabeli 3 prikazana je godišnja **Top 10** liste najtraženijih datoteka na *Sezamu*. Pošto je većina programa na listi bila prisutna u nekoliko svojih verzija, u tabeli su navedeni nazivi i direktorijumi u kojima se nalaze poslednje (najnovije) verzije tih programa.

Iz godišnje liste moglo bi se ponešto i zaključiti (ništa ozbiljno) o pogledu prosečnog Sezamovca na svet PD softvera. On (logično) voli komunikacije i arhivere, mrzi viruse ko kugu (zato je oprezan) i imao je problema sa YU slovima (a rešio ih je softverom sa *Sezama*).

Od brojki pomenimo samo da je količina PC programa skočila sa oko 5 MB krajem aprila prošle godine na gotovo 20 MB. Zbog toga i ovaj aspekt *Sezama* postaje izuzetno interesantan. Nadamo se da se u tom mnoštvu programa nalazi dosta toga zanimljivog i za vas.

SUPERSONIC SEARCH TOOL

ili skraćeno SST (u starijim verzijama programa je nosio naziv *Whereis*) najkraće bi se mogao opisati kao „još jedan file finder" i time svrstati u nebrojeno mnoštvo sličnih programa (na primer, samo na *Trickle*-u smo našli dvadesetak raznih programa te namene). Verovatno je na našim prostorima najpoznatiji takav program „Nortonov" *File Find*, ali je zato FF u NU 6.0 usled pretrpljenih izmena (kažu da je ovaj „user friendly"?) postao jedan od najneupotrebljivijih programa koje smo imali prilike da vidimo (na svu sreću, zadržali su i staru verziju, preimenovanu u *File Locate*).

Uz najstandardniju mogućnost traženja datoteka po imenu, SST uvodi i veliki broj opcija pri pretraživanju i drugih pogodnosti, a uz to je i vrlo brz. Podržani su filteri po atributima, rasponu dužina datoteka i datumskom (vremenskom) periodu. Uz standardne DOS ove džoker znake (*wildcards*) * i ?, uvedeni su još neki koji omogućavaju razne akrobacije (kao da je reč o standardnom pretraživanju stringova) prilikom zadavanja naziva datoteka za traženje. Moguće je i traženje stringova unutar opisa datoteka u 4DOS-u.

Možete navesti i DOS komandu (ili program) koji će se izvršiti sa imenima pronađenih datoteka (onih koje zadovoljavaju zadate uslove) kao argumentima. Možete ih obrisati, kopirati, pregledati, arhivirati, promeniti direktorijum u onaj gde je pronađena datoteka... ili šta vam već padne na pamet. Na primer, sve BAK datoteke ćete obrisati sa

SST *.bak -v -n

Tabela 2 Novo u decembru i ukupno

direktorijum	novo broj	novo dužina	ukupno broj	ukupno dužina
ARCHIVER	2	262 022	16	1 101 896
COM	6	722 867	30	3 029 071
DEMO	0	0	9	897 209
DRIVERS	2	112 882	15	1 541 531
EDITOR	1	135 168	13	925 814
FUN	4	531 200	11	775 099
GRAPHICS	3	380 277	11	984 992
INFO	5	599 166	8	1 084 232
MISC	2	336 624	20	2 125 742
PROGRAM	2	256 875	17	1 834 366
SYSTEM	0	0	13	982 871
TEST	1	51 915	11	831 554
UTILITY	7	332 454	33	1 379 019
VIRUS	0	0	8	595 056
WINDOWS	4	508 059	7	912 643
YU	0	0	9	134 992
Ukupno	39	4 230 519	232	19 146 087

Evo još jedne sjajne primene:

SST -d *.doc -s5 -c"copy &f a:\"

će uraditi bekap (na disketu A:) svih DOC datoteka menjanih u zadnjih pet dana.

Ukoliko želite samo da pogledate spisak traženih datoteka (ovo ćete ipak, i pored svih opcija, najčešće tražiti), imate izbor raznih oblika ispisa.

Nama je izuzetno korisna mogućnost SST-a da po disku traži sve duplirane datoteke (sa istim imenima). Obično se nakon upotrebe te opcije iznenadimo koliko nam raznih kopija istih stvari zauzima (uvek potreban) prostor.

I na kraju, sve opcije pretraživanja su omogućene i za datoteke koje se nalaze unutar arhiva! Podržani su svi standardni formati: ARC, ARJ, LZH, ZIP.

Pretpostavljamo da vam je bar neka (ili većina) od ovih silnih mogućnosti nekad zatrebala. Mi smo svojevremeno, posle samo nekoliko minuta isprobavanja, SST oduševljeno stavili u direktorijum rezervisan za neophodne korisne alatke. Verujemo da ćete to uraditi i vi.

Autor SST-a je Kejt Ledbeter (Keith Ledbetter). Program se distribuira kao „shareware" (bez ograničenja u odnosu na registrovanu verziju), a iznos registracije je 15 USD. Na Sezamu ga možete naći kao

\ibmpc\utility\sst_53a.zip

Tabela 1 TOP 10 datoteka – decembar 1991.

	direktorijum	datoteka		opis
1	\COM	sor230	zip	Sezam Off Line Reader - Verzija 2.30
2	\IBMPC\YU	hyutil20	zip	Hercules YU utilities v2.0 (dzakic)
3	\IBMPC\VIRUS	clean84	zip	Uklanja viruse koje Scan nađe
4	\IBMPC\ARCHIVER	arj222	exe	ARJ arhiver (v 2.22)
5	\IBMPC\ARCHIVER	pkz193a	exe	PkZIP arhiver verzija 1.93a (beta)
6	\IBMPC\VIRUS	scanv84	zip	McAfee virus detektor
7	\COM	tm217b	arj	Telemate v 2.17b
8	\IBMPC\SYSTEM	boot142	zip	Izbor konfiguracije (config & autoexeca) prilikom boot
9	\IBMPC\UTILITY	snippr24	zip	Kopiranje dela ekrana na štampač, file ili keybd buffer
10	\IBMPC\ARCHIVER	arjm101	zip	Shell za rad sa ARJ arhiverom

Tabela 3 Najtraženiji programi u 1991.

	direktorijum	datoteka		opis
1	\IBMPC\VIRUS	scanv84	zip	SCAN: McAfee virus detektor
2	\IBMPC\VIRUS	clean84	zip	Clean: uklanja viruse koje Scan nađe
3	\COM	sor230	zip	SOR: Sezam Off Line Reader
4	\IBMPC\ARCHIVER	arj222	exe	ARJ arhiver
5	\COM	dsz1214	zip	DSZ (ZModem) file transfer protokol
6	\IBMPC\YU	hyutil20	zip	Hercules YU utilities (dzakic)
7	\IBMPC\VIRUS	vshld84	zip	VShield: rezidentni Scan
8	\IBMPC\FUN	dash.zip		Boulder dash (VGA)
9	\IBMPC\ARCHIVER	shez64	zip	SHEZ: shell za rad arhiverima
10	\IBMPC\YU	yufont.zip	YU	slova na VGA

Decembar 1991. pokazao je blagodet rada na 16 linija – mnogo više poziva, poruka, datoteka, chat-a... Konferencije SEZAM i FORUM su „prepunjene" velikim brojem poruka pa smo otvorili nove, a bilo je i novogodišnjih čestitki svih vrsta u ogromnom broju.

Kada je reč o bilansu, tokom januara smo „opslužili" 20.426 poziva, komunicirali sa korisnicima 4.216 časova tj. 176 dana (obzirom da decembar ima 31 dan, u proseku je bilo neprekidno zauzeto 5.16 nodova što znači da se Sezam barem u dnevno vreme uglavnom dobija bez čekanja) od čega je u chat-u provedeno 1135 časova, tj. 142 radna dana (i radne noći). Korisnici su sa Sezama download-ovali 655 megabajta datoteka, dok je za isto vreme sistemu poslato oko 43 megabajta.

Što se stvari „koje ostaju" tiče, tokom decembra u javne konferencije je prispelo oko 7.800 poruka (6.6 megabajta teksta) i 3.6 megabajta datoteka priključenih uz njih, 2.900 poruka u grupama (1.4 megabajta teksta) a doživeli smo i pravi bum privatne pošte – 8.700 poruka odnosno 3.15 megabajta teksta, ne računajući priložene datoteke. Ovoliku količinu privatnih poruka (koja nas je naterala da menjamo softver jer Sezam do skore nije mogao da „podnese" više od 16.384 privatne poruke, a Ugovor nas obavezuje da privatnu poštu čuvamo barem dva meseca) objašnjavamo novogodišnjim praznicima i čestitkama koje su korisnici jedan drugom upućivali.

Deo tih čestitki stigao je i u konferenciju **SEZAM** koja se približila granici od 5.000 poruka, pa smo je pred same praznike zatvorili za upis, preimenovali u SEZAM.1 a otvorili novu konferenciju SEZAM namenjenu diskusiji o našem sistemu. Za rekordan broj poruka u ovoj konferenciji nisu krive samo čestitke, nego i žestoka rasprava o jednoj novoj naredbi uvedenoj na zahtev većeg broja korisnika – IGNORE nalaže sistemu da ubuduće ignoriše privatne poruke i poruke u konferencijama koje određeni korisnik (ili grupa korisnika) šalje. Niko nije sporio da je ova naredba potrebna, ali se postavilo pitanje da li ignorisani treba da vidi ko ga ignoriše ili ne. Bilo je dosta razgovora i o SORT-u, kao i o problemima novih korisnika Sezama koji pokušavaju da napuste menije i pređu u komandni mod.

U konferenciju **RAČUNARI** nije stiglo previše poruka – decembarski broj našeg časopisa je izašao na vreme i uglavnom je povoljno primljen, naravno uz uobičajene komentare na račun naslovne strane i tehničkog izgleda. Pred kraj godine na Sezamu je publikovana statistika „Računara" za 1991. pa je bilo diskusije i o tome, naročito oko (relativnog) sniženja cene časopisa, povremenih kašnjenja i tome slično.

U konferenciji **KOMUNIKACIJE** bilo je dosta poruka, kao i prethodnih meseci. Diskutovalo se o problemima sa pojedinim modemima i COM portovima, programu *Desk Link* i novom *Telemate*-u, maksimalnim brzinama *download*-a koje se mogu postići sa MNP emulacijom i koje se realno (ne)postižu pri razgovoru sa Sezamom, beta verziji PKZIP-a 2.0 koja je pristigla na Sezam, pronalaženju nečije *e-mail* adrese na osnovu imena, *chat*-u preko *relay* servera Bitnet-a, slanju tekstualnih datoteka na VAX protokolom ZModem... Otvorena je i tema *radio* posvećena radio amaterima, a najveći broj poruka bio je odgovor na našu želju da formiramo korektnu listu svih BBS-ova u Jugoslaviji. Posle mnogo truda, ispravki i dopuna formirana je lista koja se može preuzeti iz direktorijuma NOVOSTI.

Ukupno 400 poruka u konferenciji **PC HARD** znači da je mnogim korisnicima bila potrebna pomoć. Glavni problemi bili su sa matičnim pločama koje su zaštićene šifrom koju vlasnik ne zna, zatim video karticama koje se teško prilagođavaju bržim magistralama, SCSI diskovima koji ne rade sa nekim programima, kao i mrežnim karticama koje se „guše" u kombinaciji sa najobičnijim Herkulesom. Diskutovalo se i o novim 80486 pločama na 33 megaherca, kao i o njihovim performansama u poređenju sa „klasičnim" 386/33/64K kombinacijama.

Najviše poruka u konferenciji **PC OS** je, što se da i očekivati, pristiglo na temu *ms-dos*, a bilo je reči uglavnom o raznim mogućnostima prevazilaženja granice od 640 K i problemima pri njihovoj primeni. U okviru teme *unix* najviše je bilo reči o komunikacijama pre svega o sistemu *uucp*. Glavni „ekcesi" teme *windows* bili su fontovi sa YU slovima, a govorilo se i o tome kako konfigurisati *Windows* i DOS kada je sistem „tesan" sa memorijom. Što se teme *4dos* tiče, dobili smo prva utiske o verziji 4.0 – dosta korisnih novosti, ali na žalost i problem sa *hot key* kombinacijama koji proističe iz veze 4DOS-a i DOS-a 5.0. Tema *mreže* se uglavnom bavila *Novell*-om.

Izgleda da programeri ne mare za hladnoću, pa je konferencija **PC PROG** u decembru bila aktivnija nego ikada. Posebno nas je iznenadio veliki broj poklonika bejzika, koji su digli glas protiv svih onih koji su tvrdili da je ovaj nekada tako popularni jezik mrtav, što je rezultiralo veoma zanimljivom diskusijom o različitim verzijama bejzik interpretera i kompajlera za PC. Paskal programeri su u decembru imali najviše problema koji su, uz pomoć kolega sa Sezama, uglavnom brzo rešavani. Problemi su bili u prenosu grafičkih ekrana na štampač, čitanju tekstualnih datoteka (poznato je da *Turbo Pascal* ne dozvoljava string duži od 255 karaktera), čitanju i pisanju sektora na disk i rezidentnim programima koji se, mada se to retko radi, mogu pisati i na višim jezicima. Što se *Clipper*-a tiče, najviše se diskutovalo o knjigama za početnike u ovoj oblasti, zatim su iskusniji „klipereši" rešavali problem *backup*-a podataka, pa se rodila i ideja o pisanju posebnog arhivera za tu namenu. C programeri su nastavili diskusiju iz novembra o *Lex*-u i *Yacc*-u, čemu je doprineo i zanimljiv članak o ovim alatima u novembarskim „Računarima" autora Igora Milosavljevića (*igor.mil* na Sezamu). Kroz diskusiju je rešen i problem kompajliranja koda proizvedenog *Lex*-om na MSC 6.0 kompajleru.

Od dosta interesantnih tema u konferenciji **PC USER** izdvajamo pre svega mnoštvo poruka posvećenih programu *Stacker* koji bi, po onome kako ga reklamiraju, trebao da napravi čuda uvećavajući kapacitet hard diska putem komprimovanja podataka na njemu i raspakivanja u letu... to jest pri učitavanju. Kao što to obično biva, *Stacker* je na Sezamu stekao oduševljene pristalice ali i protivnike, to jest one koji su malo oprezniji. Bilo je naravno i drugih interesantnih tema. Od tekst procesora i editora pažnju su izazvala prva (prilično bagovita) iskustva sa *WordPerfect*-om for *Windows*, kao i novi *Brief*, a posebno njegova veza sa MSC PWB-om. Bilo je i (već standardnih) diskusija o grafičkim programima i konverziji datoteka raznih formata, fontovima, raznim uslužnim programima, drajverima za tastaturu i ostalim (vrlo brojnim) korisničkim temama.

Decembar je, što se **ATARI ST** konferencije tiče, bio pre svega u znaku emulatora *ZX Spectrum*-a – povezivanje ST-a i *Spectrum*-a preko RS232 i MIDI-ja i prebacivanje „starih dobrih" igrica na diskete bile su glavne teme. Osim toga, pričalo se o *Macintosh* emulatoru, igrama koje se mogu igrati preko modema i mnogim programerskim temama. Novi komunikacioni program *Tripido* koji je u prvo vreme izgledao bolji od svega što do danas postoji na ST-u je ipak razočarao, pa su ataristi ostali verni *Flash*-u, *Uniterm*-u i *Rufus*-u. Sezamov „projekat" instaliranja YU fonta i tastaturne tabele je ponovo „oživeo", a magazini STReport, ZNET i ST Digest su prebačeni iz ove konferencije u konferenciju NOVOSTI.

Što se **FORUM**-a tiče, znate već: situacija je dovoljno dramatična da limit od 5.000 poruka po konferenciji zahteva otvaranje novog FORUM-a na svakih par meseci pa smo i u Novu godinu ušli sa novim FORUM-om, petim po redu. Čini se ipak da je u decembru bilo nešto manje poruka o ratu nego ranijih meseci, a glavna diskusija se preselila u temu *devojke* u vidu žestoke svađe zbog jedne podrugljive pesmice koju je napisala jedna korisnica. Posle nekoliko dana glavni akteri su se povukli iz rasprave, tako da su je nastavili oni kojima je ta pesmica samo bila povod da svode na drevni red neke svoje „stare račune".

Konferencija **EXTRA** „ponovila" se temom *misterije* u okviru koje se diskutuje o raznim neobjašnjivim ili neobješnjenim događajima i literaturi iz ove, očito veoma popularne oblasti. Dosta poruka je bilo i u vezi naučne fantastike, horoskopa, simulatora leta (i vazduhoplovstva uopšte).

Konferencija **CIVILIZACIJA** je uhvatila određeni ritam bez naglih promena, kako u pogledu kvaliteta tako i kvantiteta priloga, što ju je karakterisalo u samom početku. To je dobar znak, pogotovo za konferenciju koja ima ambicije da bude više obrazovna nego polemička. Najaktivnije teme bile su *religija* i *književnost*. Razgovaralo se o svemu i svačemu, počev od arhitekture renesansnih crkvi, egipatske i antičke mitologije, pa preko kineskih i zen priča, sve do originalnih književnih doprinosa.

Za dobro raspoloženje na Sezamu tradicionalno je zadužena konferencija **VICEVI** u koju osim domaćih pristižu i vicevi sa stranih paketnih mreža. Ukoliko autor ovog teksta može da kaže da mu se već više meseci ne dešava da čuje neki vic koji već nije pročitao na Sezamu. Umesto priče o konferenciji VICEVI, da završimo ovaj tekst sa dva kratka vica postigla tokom decembra koja se visoko kotiraju u glasanju za vic meseca. Prvi (aktuelan, američki): „Znate li kako je Fredi Merkuri dobio AIDS? It's a kind of Magic!" Drugi (da se malo podsetimo na prošlost): Razgovaraju dva Rusa u bistrou posle rada: „Radim u fabrici bicikala... baš bih voleo da imam jedan" „Pa dobro, zašto nešto ne preduzmeš?" „Preduzimam, svaki put kada odlazim sa posla, dipim po jedan deo..." „I?" „Pa ne znam, svaki put na kraju ispadne mitraljez!"

KAKO POSTATI ČLAN

Postupak učlanjenja počinje zvanjem Sezama. Nakon poziva broja 648-[illegible] i uspostavljanja veze (parametri 2400N8) i pozdravne maske, dobićete pitanje

Username

Na njega treba odgovoriti sa NEW (new korisnik), nakon čega vas Sezam vodi kroz proces prijave. Prvi podatak koji treba da date je vaše puno ime i prezime, a potom birate pseudonim pod kojim ćete koristiti sistem. Pseudonim je bilo kakva reč do 16 znakova koja sadrži samo mala slova i znak . (tačka) i obično se bira tako da asocira na vaše ime. Sistem će na osnovu vašeg pravog imena predložiti pseudonim koji možete ali i ne morate prihvatiti. Pre izbora pseudonima dobro razmislite, jer on postaje vaše trajno ime na Sezamu i ne možete ga više promeniti.

Sledeći korak je izbor lozinke (password) kojim se vaš pseudonim štiti – prijavljivanje sistemu je moguće samo ako poznajete lozinku koja je vezana uz vaš pseudonim.

Nakon ovih osnovnih podataka sledi izbor nekoliko raznih parametara (objašnjenje cele stvari na ekranu) a potom unos ostalih ličnih podataka – adrese, telefona i podataka o opremi koju koristite za zvanje Sezama. Ovim je prva faza gotova.

Posebnu pažnju obratite na adresu unesete korektno – ispravna adresa je jedini način da Uprava ostvari dalji kontakt sa vama kako bi se do kraja obavio proces vašeg učlanjenja.

Sledećeg radnog dana će vam biti poslato pismo u kome ćete pronaći i popunjenu uplatnicu. Na vama je da izaberete period na koji se učlanjujete i na osnovu toga na uplatnicu upišete iznos pretplate:

za 3 meseca	1000 din
za 6 meseci	1800 din
za godinu dana	3000 din

Nakon izvršene uplate, fotokopiju potvrde o uplati pošaljite na naznačenu adresu Uprave Sezama bilo poštom ili (daleko bolje) faksom. Po prijemu ove potvrde, Uprava će vam istog radnog dana odobriti pristup sistemu, čime postajete punopravni član Sezama. Istog dana će na vašu adresu biti poslato i Uputstvo za korišćenje sistema.

Ukoliko pri prvom prijavljivanju imate probleme i ne završite popunjavanje podataka korektno nije potrebno da zovete ponovo i prijavljujete se pod novim imenom. Sačekajte jedan dan pa pozovite ponovo – administratori sistema kod detektovanja nepotpunih prijava intervenišu i daju vam mogućnost da pri sledećem javljanju ponovo popunite prijavu, [illegible]

Ukoliko ste se pak prijavili pod više različitih pseudonima, [illegible] Uprava [illegible]

Ukoliko želite [illegible] da razgledate sistem na [illegible] sa DEMO. Način će se demonstracije [illegible] u kome možete pregledati kompletnu strukturu [illegible] pravog [illegible] DEMO [illegible] vreme za jednu seansu je 5 minuta.

SPECIJALNI
KOMERCIJALNO-INFORMATIVNI BLOK
• akcije • marketing • publicitet •

Poštovani prijatelji

„Računari" se štampaju isključivo na uvoznom papiru i sa uvoznim repro materijalima što se plaća devizama Međutim cene oglasnog prostora rasle su znatno sporije nego što je potrebno da bi se pariralo padu vrednosti dinara u odnosu na strane valute Zbog toga je danas realna vrednost oglasnog prostora u „Računarima" nekoliko puta manja nego samo pre godinu dana S obzirom da niže od tih vrednosti ne može da se ide cena će se od sada održavati približno na zatečenom nivou u odnosu na druge valute Aktuelna ekonomska situacija nameće nam obavezu da se **oglasi fakturišu i plaćaju pre izlaska broja iz štampe** U izuzetnim slučajevima kao što su godišnji aranžmani oglasi će se plaćati nakon izlaska broja ali uz strogo poštovanje utvrđenih rokova

Cene oglasnog prostora u „Računarima" iznose

— **1/1 poslednja strana korica**	120 000 D
— **1/1 druga strana korica**	110 000 D
— **1/1 treća strana korica**	110 000 D
— **1/1 kolor unutrašnja strana**	100 000 D
— **1/2 kolor unutrašnja strana**	60 000 D
— **1/1 crno-bela strana**	70 000 D
— **1/2 crno-bela strana**	40 000 D
— **1/4 crno-bela strana**	25 000 D
— **1/8 crno-bela strana**	20 000 D
— **mali oglas (po centimetru)**	2 000 D

ROK ZA DOGOVOR I REZERVACIJU PROSTORA je 5, a za slanje oglasa 10 u mesecu (za broj koji izlazi početkom narednog meseca) DUGOROČNIJE OGLAŠAVANJE U „Računarima" (u 6 9 ili 12 brojeva) predaja POTPUNO PRIPREMLJENIH OGLASA (sa urađenim filmovima za kolor odnosno filmom ili otiskom sa lasera za crno-beli oglas) i PLAĆANJE UNAPRED za tri ili više oglasa **donose vam odgovarajuće popuste** Molimo vas da se za dogovore i informacije javite na
Telefon 011/653 748, fax 011/648-140, modem 011/648-899
(osoba za kontakt Esad Jakupović)

Unapredite svoj stil — knjige za Prave Programere!

1) Uvod u objektno programiranje
(Turbo Pascal 5.5 i 6.0 Modula-2 C++ Smalltalk/V)
Prva knjiga te vrste kod nas i u svetu!
Izaberite svoj jezik! Saznajte zašto je najbolji
Autor Duško Savić Strana 250 1992 cena **900 dinara**

2) Turbo Pascal — naredbe i objekti
Kompletan priručnik objašnjenje svih naredbi Objekti
Autor Duško Savić Strana 240 1991 cena **550 dinara**

Poštarina **100 dinara** po knjizi Zovite ili pišite odmah
PC Program. Generala Hanrisa 20
11040 Beograd tel (011) 463 296

Ćirilica — Miroslavljeva, Swiss Tajms
YU latinica za BANNER
Posle 15 h Tel 011/506-259

IBM PC & COMPATIBLE
Izrada i ponuda programa za PC računare iz svih oblasti
NOVO — BESPLATNO Najveći izbor PC software-a (Public Domain) u zemlji! Imamo **SVE! KATALOG** Besplatni savjeti inženjera specijaliste EE Software F G Murića 31 78000 Banja Luka, tel 078/40-640

INFORMIX 4GL PRIRUČNIK Instrukcije sintaksa primjeri 180 str 1800 din Jovičić Lenjinova 28 71000 Sarajevo

Englesko-srpskohrvatski
Srpskohrvatsko engleski
REČNIK ZA PC
sa oko 40 00 reči MGA I VGA verzija YU znaci zauzima 3MB na HARD DISKU na 5.25" ili 3.5 disketama
Cena sa disketom 1.300 - poštarina
tel 031/27 832

KOMPJUTER SERVIS
— SPEKTRUM
— COMODORE 64 i periferije
— AMIGA 500
— PC
Avalska 6 (na Slaviji)
tel 011/434 287

AT
286
386
486

štampači, delovi
Programi za:
knjigovodstvo, LD, STR, ugostiteljstvo

ISPORUKA ODMAH
☎ 011/454-987
Sigma computers

Trgovinsko-knjižarsko preduzeće „ŠAHINPAŠIĆ" — Sarajevo

TOP 20 HOT LIST

1.	E. IHRIG:	COREL DRAW! 2 — Covers Versions Through 2.01 — McGraw-Hill	DM 81,98
2.	T. Sheldon:	NOVELL NETWARE THE COMPLETE REFERENCE — Covers Versions Thru 2.15 — McGraw-Hill	DM 140,21
3.	N. Johnson:	AUTOCAD THE COMPLETE REFERENCE — Covers Release 10 & 11 — McGraw-Hill	DM 105,11
4.	V. Wright:	THE BEST BOOK OF AUTOCAD — Release 11 — SAMS	DM 102,00
5.	K. Acerson:	WORD PERFECT 5.1 THE COMPLETE REFERENCE — McGraw-Hill	DM 91,65
6.	S. O'Brien:	TURBO PASCAL 6 THE COMPLETE REFERENCE — McGraw-Hill	DM 89,51
7.	C. Petzold:	PROGRAMMING WINDOWS 2ed. — Microsoft Press	DM 104,79
8.	N. Barkakati:	MICROSOFT C BIBLE — SAMS	DM 86,94
9.	N. Barkakati:	TURBO C BIBLE — SAMS	DM 86,94
10.	J. Borst:	VENTURA PUBLISHER FOR WINDOWS — Microsoft Press	DM 104,82
11.	K. Jamsa:	D O S THE NEW COMPLETE REFERENCE — McGraw-Hill	DM 93,41
12.	R. Shushan:	DESKTOP PUBLISHING BY DESIGN — VENTURA — Microsoft Press	DM 86,03
13.	C. Jenson:	HARVARD GRAPHICS THE COMPLETE REFERENCE — McGraw-Hill	DM 98,15
14.	C. Coffin:	UNIX THE COMPLETE REFERENCE — McGraw-Hill	DM 106,80
15.	S. Starkey:	PROGRAMMING IN CLIPPER — Addison Wesley	DM 74,63
16.	S. Starkey:	ADVANCED PROGRAMMING IN CLIPPER WITH C — Addison-Wesley	DM 74,63
17.	L. Omura:	LEARN CAD NOW — Microsoft Press	DM 134,82
18.	C. Matthews:	WORDSTAR PROFESSIONAL THE COMPLETE REFERENCE — McGraw-Hill	DM 98,98
19.	C. Robinson:	THE BEST BOOK OF FOX PRO — Roydon Books	DM 86,00
20.	M. Edelhart:	INTEL OFFICIAL GUIDE TO 386 COMPUTING — McGraw-Hill	DM 90,63

TRGOVINSKO-KNJIŽARSKO PREDUZEĆE „ŠAHINPAŠIĆ" p.o. — SARAJEVO Livanjska 74 Telefon 3871/212-333 274 022 Fax 3871/212-333 Žiro račun 10100-601-17697 KNJIŽARA INOZEMNE KNJIGE — SARAJEVO Sime Milutinovića 10 Tel 3871/275-574

NOVO

PC SERVIS

• Popravka svih vrsta PC računara
• Servis sa bogatim iskustvom
• Drugi menjaju, mi popravljamo
• Računar je najskuplji kada ne radi
• Vi nas pozovete, mi dolazimo



RADIMPEX • IZVORSKA 55 • BEOGRAD • TEL 543-364, 335-443

Clipper 5.0

NiSoft biblioteka

CLIPPER 5.0 - BIBLIJA

cena 20 DEM (u dinarskoj protivvrednosti na slobodnom tržištu)

Najkompletnije uputstvo za Clipper na 620 strana B5 formata (17x24cm)

- instalacija
- osnove jezika
- pretprocesorske direktive
- izrazi, naredbe, funkcije
- rad u mreži
- klase i objekti
- kompilacija i linkovanje
- podaci kojih nema ni u originalnoj dokumentaciji
- višegodišnje iskustvo grupe autora

Knjigu možete poručiti pouzećem (telefonom, telefaksom, pismom), uplatom na žiro račun (obavezno poslati kopiju uplatnice) ili poštanskom uputnicom sa punom adresom naručioca.

U pripremi dodatak sa svim izmenama za verziju Clipper 5.01

Žiro račun "NiSoft biblioteka"
62500-625-18007-80701/39004
Adresa:
"NiSoft biblioteka"
V kongresa 63/3, 18000 Niš

Tel/Fax: 018/ 44-198
Telefon: 018/712-664
018/710-923

EDICIJA KOMPJUTER BIBLIOTEKA ČAČAK
vam predstavlja sva najtraženija izdanja

1 TVRDI DISK (250 str) 450 din
Nova knjiga sve o TVRDOM DISKU
2 TURBO PASCAL 5 5 (450 str) 450 din
Nezaobilazna knjiga za korisnika
3 MS-DOS 3 3 (250 str) IV izdanje 450 din
Kompletan priručnik

CENA KOMPLETA 980 din
Cene važe do 28 02 1992 g Porudžbine slati na adresu
Edicija KOMPJUTER BIBLIOTEKA, F. Filipovica 41, 32000 Čačak, ili na telefone **(032) 32-322 ili 23-120**

YU SLOVA
LATINICA , ĆIRILICA
ZA SVE *ŠTAMPAČE*
011/347-509, 403-229

Diskete ESCOM SONY svih formata
Tel 034/34-619 i 034/63-011

KOMPJUTEROM DO ZARADE
pomoću KLUBA POSLOVNIH KOMPJUTERAŠA i Vi možete do bro zaradili obavljanjem honorar nih kompjuterskih poslova! Za bes platna obaveštenja poslati adresi rani koveral sa markom
Nenad Stoiljković 21000 **Novi Sad** Put part zanskih baza **tel 021/397-743**

ELHARD
25238 KULA Tel. 025-722-740
V. VLAHOVIĆA 46 Fax. 025-723-433

Prodajemo kompjutere u konfiguracijama po Vašoj želji i u delovima

Garancija 12 meseci
Imamo

AT 286/16 MHz
AT 286/20 MHz
AT 386 i AT 486

Neprekidno napajanje
500 i 1000 W

Miševi

Handy scanner
400 dpi sa OCR
programom

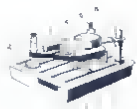
Za cene se obratite na naš telefon
od 6 00 do 18 00 sati svakog dana
Subotom od 6 00 do 12 00 sati

Nudimo Vam i veliki izbor
dodatne opreme za Vaše ljubimce

Kod kupovine većih količina dajemo popust

NOVO NA TRŽIŠTU!!

Aparat MAXIPRINT obnavlja Vaše trake Na zaboravite da sa 10 obnavljanja MAXIPRINT postaje besplatan Ne bacajte korišćene traka



Kupite MAXPRINT i uštedite 95% od tro škova koje plaćate za trake

Radi sa svim trakama za štampače Ako ne verujete u njegovu efikasnost pošaljite jed nu traku i mi cemo je besplatno osvežiti

Javite se na MIKROSAM 97500 Prilep Đura Đorđeski 5 ili na teleton 096/34-072 096/29 965 096/30-007

MAXIPRINT dobijate poštom „pouzecem"

M I K R O S A M
ĐURE ĐORESK 5 97500 PRILEP JUGOSLAVIJA TEL 096 34 972 30 097 FAX 096) 34 012
NAJNIŽE CENE OPREME I USLUGA
JEFTINIJI SMO OD SVAKE DRUGE PONUDE ZA 5%
LIČNI RAČUNARI SVAKA KONFIGURACIJA
ŠTAMPAČI SVI TIPOVI I FORMATI
LASERSKI ŠTAMPAČI
POVEZIVANJE U LOKALNU MREŽU
OPTIČKO ARHIVIRANJE DOKUMENTACIJE
FAKS APARATI SVIH VRSTI
REPROMATERIJALI DISKETE TRAKE ZA ŠTAMPAČE HARTIJA
PROGRAMI ZA SVE VRSTE POSLOVNIH AKTIVNOSTI
GRAFIČKA PRIPREMA ZA ŠTAMPANJE
UZ MIKROSAM VI STE OSUĐENI NA USPEH !!!

BESPLATNI PROGRAMI!

izaberite izmedu 7000 naslova

APP SYSTEMS
Zadarska 6 Beograd
tel: 011-630-167 tel/fax: 011-635-664

KUPUJTE PROVERENE DISKETE PUNE BESPLATNIH PROGRAMA!

Svaka disketa sadrži najmanje 700Kb kompresovanih programa!

Kompleti disketa za Clipper, TurboPascal, C, BASIC, Windows 3.0, AutoCAD, Fontovi, Sistemski programi, Laserski štampači itd.
IGRE IGRE IGRE IGRE

Besplatan katalog sa preko 7000 naslova
Isporuka pouzećem ili lično

OTKUP POLOVNE RAČUNARSKE OPREME!
PRODAJA RAČUNARA STARO ZA NOVO!
PRODAJA REPARIRANIH RAČUNARA!

T.P.P. PARTNER i V.D. SOFT

37000 KRUŠEVAC, CELJSKA 11

TEL/FAX 037-25293

RADNO VREME:
OD 8—14h i OD 18—20h



AT-286/16MHz VGA MONO

NUDIMO IZUZETNO POVOLJNO

RAČUNARE 286, 386

INSTALACIJU NOVELL MREŽA,
UNIX, XENIX OPERATIVNIH SISTEMA

ŠTAMPAČE „STAR" I „EPSON"

PROGRAME:

FINANSIJSKO KNJIGOVODSTVO
TRGOVINSKO POSLOVANJE
MAGACINSKO POSLOVANJE
PROGRAMI ZA S.T.A.
STAMBENE ZADRUGE
ZANATSKE ZADRUGE
UPRAVE PRIHODA
I GEODETSKE UPRAVE.
KUPCI DOBAVLJAČI

MM COMPUTERS
Preduzece MZ 11000 Beograd
Dubljanska br 70 Tel 011/434 812
Fax 011 450 471

ŠTAMPAČI

EPSON

LX-400 9 igličnI A4 format 180 znakova/s
LX-850 9 igličnI A4 format 200 znakova/s
FX-850m 9 igličnI A4 format 300 znakova/s
LX 1050 9-igličnI A3 format 200 znakova/s
FX 1050 9-igličnI A3 format 300 znakova/s
LQ-450 24 igličnI A4 format 180 znakova/s
LQ 850+ 24 igličnI A4 format 300 znakova/s
LQ-1050+ 24 igličnI A3 format 300 znakova/s
LQ-860 24 igličnI A4 format kolor 300 znakova/s
LQ 1060 24 igličnI A3 format kolor 300 znakova/s
LQ 2550 24 igličnI A3 format kolor 400 znakova/s
DFX 5000 „heavy duty" 9 igličnI 533 znakova/s A3 format
DFX 8000 „heavy duty" 9-igličnI A3 format 1066 znakova/s
EPL 7500 laserski štampač sa originalnim Adobe PostScriptom
GT-4000 color scanner
GT 6000 color scanner

Roland DIGITAL GROUP

DXY 1100	*A3 PLOTER*
DXY 1200	*A3 PLOTER*
DXY 1300	*A3 PLOTER*
DPX 2500	*A2 PLOTER*
DPX-3500	*A1 PLOTER*
GRX 300	*A1 „ROLL" PLOTER*
GRX 400	*A0 „ROLL" PLOTER*
DPX-4600	*A0 PLOTER*
LTX 100	*A3 TERMIČKI PLOTER*
LTX 120	*A3 TERMIČKI PLOTER*
LTX-321	*A1 TERMIČKI PLOTER*
LTX-420	*A0 TERMIČKI PLOTER*
CAMM 1	*PNC 1000 PLOTER REZAČ*
CAMM-1	*PNC 1800 PLOTER REZAČ A0 FORMAT*
STIKA	*REZAČ SA UGRADENIM SKENEROM*

SKETCHMATE A4 PLOTER
ROLAND PERA ZA PLOTERE

EIZO

EIZO 4050	*14" MONOHROMATSKI MONITOR*
EIZO 6500	*21" MONOHROMATSKI MONITOR*
EIZO 9060S	*14" KOLOR MONITOR*
EIZO 9070S	*16" KOLOR MONITOR*
EIZO T560	*17" KOLOR TRINITRON MONITOR*
EIZO 9400i	*20" KOLOR MONITOR*
EIZO T660	*20" KOLOR TRINITRON MONITOR*
EIZO MD 808	*GRAFIČKA KARTICA*
EIZO MD 810	*GRAFIČKA KARTICA*
EIZO MD 812—60	*GRAFIČKA KARTICA*

ISPORUKA ODMAH
GARANCIJA GODINU DANA
POSEDUJEMO ĐODATNI I POTROŠNI MATERIJAL
OVLAŠĆENI SERVIS
INFORMACIJE NA TEL 011 434 812
FAX 011 450 471

Microsoft
NOVELL
ASHTON·TATE
Fox Software
Peter Norton
WORDSTAR
P.O. BEOGRAD
Glasinačka 31/52
Tel. 011/488-5472
BORLAND
Lotus
AUTODESK

PC Club
PCCLUB d.o.o.
računarski inženjering
i vršenje usluge u zemlji i inostranstvu
Beograd, ul. 21. divizije 44
telefon 011/444-30-79 telefax 011/422-199

286 386 486

PRODUKTI NAJBOLJEG KVALITETA KROJENI PO VAŠIM POTREBAMA

IBM PC RAČUNARI, SCO UNIX, POSLOVNE APLIKACIJE, MREŽE

MI VAM NUDIMO MAKSIMALNI KVALITET I MINIMALNE CENE, TEHNIČKU POMOĆ, SERVIS, POSLOVNU SARADNJU I PRIJATELJSTVO U SVETU TEHNIKE

MODEL	MONO	RAM	SVEGA
	DEM		DEM
ISC325 – 386SX-25 MHz	2.510	2Mb	3.310
ISC333 – 386DX-33 MHz/128k Cache	3.540	4Mb	4.340
ISC420 – 486SX-20 MHz/128k Cache	4.075	4Mb	4.875
ISC433 – 486DX-33 MHz/256k Cache	5.390	8Mb	6.190

Svi modeli sadrže Baby kućište, HD 40Mb, FD 1.2Mb 2Ser + 1Par Port, tastaturu, monohromatski monitor

DOPLATE:	DEM
HD 80Mb IDE	500
HD 105Mb IDE	900
HD 340Mb SCSI	4.100
1Mb RAM	150
FD 1.44Mb	180
Tower kućište	160
EPSON FX-1050	1.775

ŠIROK IZBOR I DRUGE RAČUNARSKE OPREME, POSEBNO ZA KONFIGURACIJE SERVERA I RADNIH STANICA U MREŽI

INTELIGENTNI MULTIPLEKSER (SPECIALIX), TERMINALI (PC Term, VT220, ANSI emulacija)
SISTEMSKI SOFTVER. SCO Unix, DOS, X-Windows, VPIX, TCP/IP, NFS

APLIKATIVNI SOFTVER:
Upravljanje poslovanjem, projektovanje, upravljanje procesima

ISPORUKA ODMAH!

Preduzeće ISC Beograd, Tel. 011/505-785

MB SOFT
NOVI SAD

Program za personalne računare koji će te koristiti svakodnevno!

Standardni englesko-srpski srpsko-engleski

REČNIK ZA PC RAČUNARE verzija 2.0

- *PREKO 90.000 REČI IZ SVIH OBLASTI*
- *ELEKTRONSKI PREVODILAC*
- *GRAMATIKA*



- TRENUTNO nalaženje reči na srpskom ili engleskom jeziku
- mogućnost NEOGRANIČENOG dopunjavanja novim rečima
- "SOFT SEEK" - inteligentna pretraga
- učitavanje engleskog ili srpskog teksta u ASCII formatu i njihovo prevođenje
- GRAMATIKA engleskog jezika
- "ON LINE" help
- VELIČINA PROGRAMA 3.8 MB
- 5 disketa i priručnik

Porudžbine ☎ (021) 612-752 i 368-476

BEST OF TK

Dr Mirjana i dr Miroslav Nikolić
EXCEL 3.0
214 str. **Broj knjige 073**
Redovna cena 1 200 d
Klupska cena 1 080 d

Dr Mirjana i dr Miroslav Nikolić
MS DOS SHELL & MS WINDOWS
156 str. **Broj knjige 076**
Redovna cena 920 d
Klupska cena 828 d

Aleksandar Radovanović
PC MODEMSKE KOMUNIKACIJE
Broj knjige 057
Redovna cena 870 d
Klupska cena 783 d

UVOD U GW BASIC
Broj knjige 086
Redovna cena 920 d
Klupska cena 828 d

LOTUS 1—2—3 U 10 LEKCIJA
Broj knjige 067
Klupska cena 990 d

Juro Studer
FRAMEWORK III
234 str. **Broj knjige 062**
Klupska cena 882 d

MS DOS UZ PODRŠKU TURBO PASCAL-a
Broj knjige 084
Klupska cena 756 d

CBIWRITER Verzije 3 i 4
Broj knjige 066
Klupska cena 891 d

PRISTUPNICU ZA KLUB MOŽETE POPUNITI I BEZ PORUČIVANJA KNJIGA IZ OVOG OGLASA

KLUB TEHNIČKE KNJIGE

Porudžbenica i pristupnica pošaljite na adresu.
KLUB TEHNIČKE KNJIGE, Beograd, Vojvode Stepe 89,
tel. 011/491-931, 492-075, telefax 473-442

PRAVILA KLUBA I POVLASTICE

PRISTUPNICA I PORUDŽBENICA 05/92

” ПЕРИХАРД – ИНЖЕЊЕРИНГ ”

PERI HARD® INŽENJERING

HEWLETT PACKARD
Authorized Dealer

RIBONI

ZA SVE

VRSTE ŠTAMPAČA

> EPSON
(DFX 5000,
LQ 550, 1000, 1500,
LX 80, 400, FX 1000, 1050, ...)
> FUJITSU
(DL 1100, 2400, 5600, 3300, 8800,
DX 2100, 2200, 2300, 2400,
M 304X, 3040, 3041, ...)
> IBM
(1403, 4201, 4202, 4207, 4224, 4234, ...)

Ostali potrošni materijal:
diskete, strimer-trake,
magnetne trake itd.

Београд, Ивана Милутиновића 24
тел. (011) 436-019, 432-383, 432-319
факс 435-513

DeskJet 500 i 500c



КОЛОР

ЛАСЕРСКИ ШТАМПАЧИ,
СКЕНЕРИ, ПЛОТЕРИ . . .
РЕЦИКЛИРАЊЕ ТОНЕР КАСЕТА

(ОТКУПЉУЈЕМО ПРАЗНЕ ТОНЕР КАСЕТЕ И КАСЕТЕ ЗА DeskJet 500)

ЗАШТИТНИ ФИЛТЕРИ ЗА МОНИТОРЕ



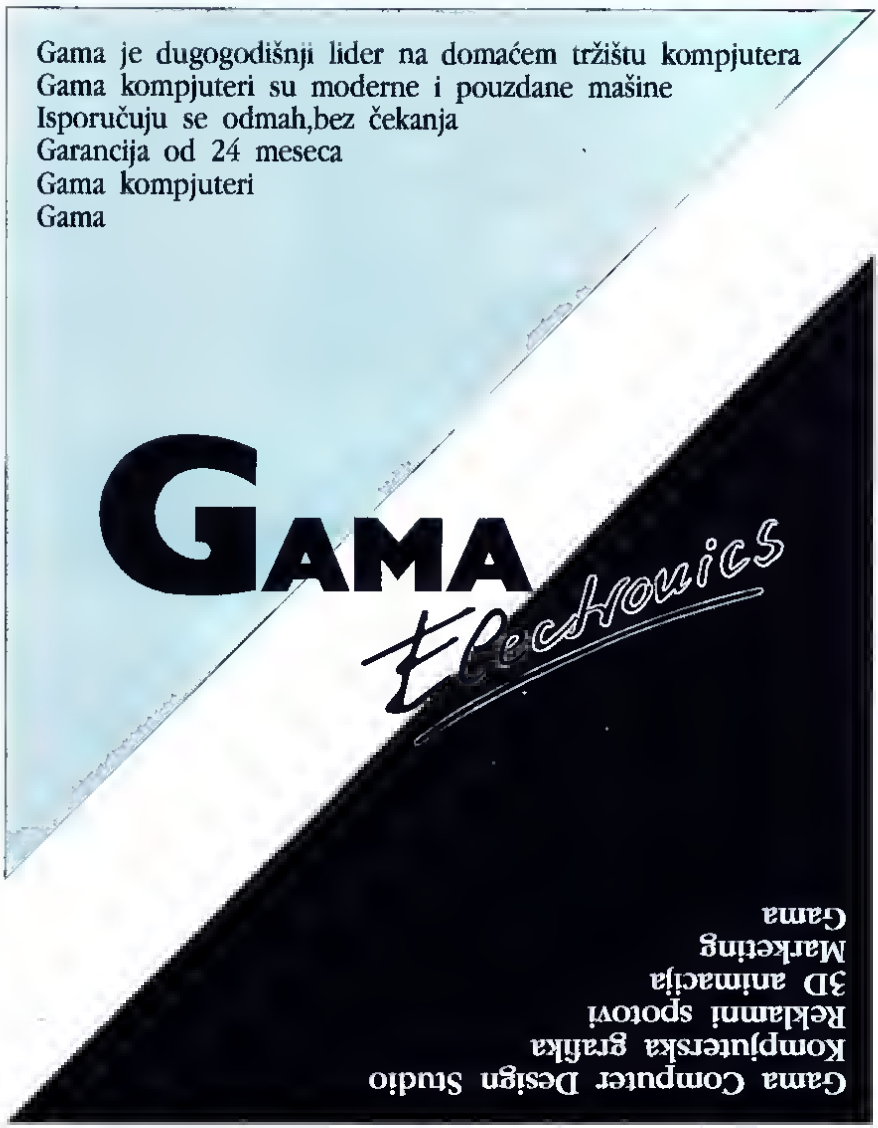
POTROŠNI MATERIJAL U VELEPRODAJI

PROTEUS

PRIVATNO PREDUZEĆE ZA PROMET TEHNOLOGIJAMA I USLUGAMA

71000 Sarajevo
Tršćanska 7
Tel: 071/201-481
Tel: 071/201-482
Fax: 071/201-480

AKO TRAŽITE MAKSIMUM



Gama Electronics d.o.o,Mišarska 11,Beograd Tel:33-22-75,33-94-94,Fax:335-902
radno vreme:ponedeljak - petak od 9 do17h
Gama Electronics Trade Handels GmbH,Tel 99/49/89/577-209,Fax 99/49/89/570-4379

ComTec d.o.o.

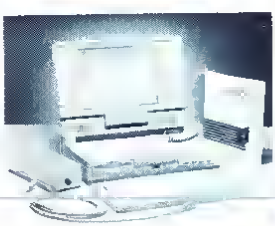
Preduzeče za informatički inženjering
i automatizaciju
Novi Sad, Vojvodanskih hrigada 7/IX
Telefon/Fax: 021/22.258

LAPTOP - NOTEBOOK





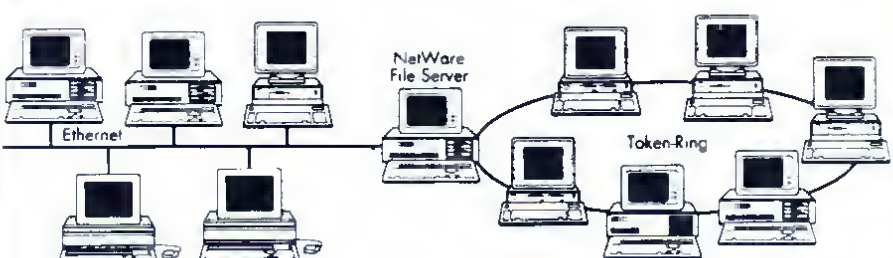
VERIDATA EL 386
samo 2kg!



FLYER 286 FH
FLYER 386 FH

VERIDATA LP 286
VERIDATA LP 386

NOVELL



ComTec d.o.o. je autorizovani prodavac *NOVELL* proizvoda
Nudimo Vam i standardne AT 286 /386/486 računare i š irok izbor aplikativnog softvera prilagodenog radu u mreži

Dileri, javite se! Povoljan rabat!

PC AT 286, 386-SX, 386, 486
ISPORUKA ODMAH

MODEL	SC 2016	SC 2025!!!	SC 3025 SX	SC 3040 CACHE 64 Kb	SC 4033 CACHE 64 Kb
PROCESOR	INTEL 286	INTEL 286	INT 386 SX	A MD 386	INTEL 486
CHIP SET	HEADLAND	HEADLAND	HEADLAND	FDRTEX	ETEC
RADNI TAKT	16 MHz	25 MHz !!!	25 MHz	40 MHz	33 MHz
LAND TEST	21 MHz	33 MHz !!!	33 MHz	62 MHz	150 MHz
KUĆIŠTE	BABY AT	BABY AT	BABY AT	BABY AT	BABY AT
NAPAJANJE	200 W	200 W	200 W	200 W	200 W
RAM	1 Mb	1 Mb	1 Mb	4 Mb	4 Mb
MAX RAM	4 Mb	8 Mb	8 Mb	32 Mb	32 Mb
FLOPPY	1 2 Mb	1 2 Mb	1 2 Mb	1 2 Mb	1 2 Mb
HARD DISK	NEC D 3735	NEC D 3735	NEC D 3735	NEC D 3735	NEC D 3735
	42 Mb	42 Mb	42 Mb	42 Mb	42 Mb
HD/FD CONT	AT BUS	AT BUS	AT BUS	AT BUS	AT BUS
	INT 1 1	INT 1 1	INT 1 1	INT 1 1	INT 1 1
CORETEST	T/T 7 ms	T/T 7 ms	T/T 7 ms	T/T 7ms	T/T 7 ms
	500 K/S	500 K/S	500 K/S	500 K/S	500 K/S
SER INT	2* RS 232	2* RX-232	2* RS 232	2* RS 232	2* RS 232
PAR INT	2* CENTR	2* CENTR	2* CENTR	2* CENTR	2* CENTR
TASTATURA	101 TAST	101 TAST	101 TAST	101 TAST	101 TAST
GRAF KART	HERCULES	HERCULES	HERCULES	HERCULES	HERCULES
MONITOR	AMBER ili	AMBER ili	AMBER ili	AMBER ili	AMBER ili
14"	CRNO-BELI	CRNO-BELI	CRNO-BELI	CRNO-BELI	CRNO-BELI
CENA DEM	**1 650 DEM**	**1 800 DEM**	**1 990 DEM**	**3 350 DEM**	**4 400 DEM**

14" VGA COLOR MONITOR		(ZVEZDICOM SU OZNAČENE
+ VGA CARD 512 Kb 1024*768	600 DEM	* DODATNE U ODNOSU NA
19" VGA MONITOR 1280 * 1024	2 800 DEM	* OSNOVNE KONFIGURACIJE)
HD 115 Mb NEC D3761, 18 ms	500 DEM	
HD 125 Mb MAXTOR	600 DEM	*
HD 210 Mb CONNER 15 ms	1 150 DEM	*
HD 330 Mb S C S I	3 000 DEM	*
TOWER KUĆIŠTE 220 W	100 DEM	*
1 Mb RAM	150 DEM	
IIT 287 10 MHz KOPROCESOR	200 DEM	(100—300 % BRŽI OD INTELA 287)
IIT 287 20 MHz KOPROCESOR	250 DEM	(100—300 % BRŽI OD INTELA 287)
IIT 387-25 MHz KOPROCESOR	300 DEM	(100—300 % BRŽI OD INTELA 387 SX)
IIT 387 40 MHz KOPROCESOR	500 DEM	(100—300 % BRŽI OD INTELA 387)
FLOPPY 1 44 Mb 160 DEM 1 2 Mb	200 DEM	
MIŠ GM 6 (RS 232)	60 DEM	
EPSON LX 400 (A4 9 PINA)	pozovite nas!	(PRINTERI SU SA UGRAĐENIM YU
EPSON LQ 200 (A4 24 PINA)	pozovite nas!	ZNACIMA ŠZČĆD I KABLOM ZA
EPSON LQ 450 (A4 24 PINA)	pozovite nas!	VEZIVANJE SA RAČUNAROM)
EPSON LD 550 (A4 24 PINA)	pozovite nas!	
EPSON FX 1050 (A3 9 PINA)	pozovite nas!	
EPSON LX 1050 (A3 9 PINA)	pozovite nas!	

UPLATA JE U DINARSKOJ PROTIVVREDNOSTI
DEM NA TRŽIŠTU U BEOGRADU
NAČIN PLAĆANJA: 100% AVANS — ISPORUKA ODMAH
ŽIRO RAČUN: 60815-601-64787
FAX: 011/345-126

PREDUZEĆE „ŠUTLIĆ“
& „MIKRO DIZAJN“

KOSOVSKA 32 / I SPRAT, 11000 BEOGRAD
RADNO VREME: 9—17 h, SUBOTOM I NEDELJOM NE RADIMO

TEL: 011/332-607 FAX 011/345-126

GARANCIJA 12 MESECI, SERVIS I ODRŽAVANJE
BEOGRAD 01. 02 '92.

CENE OPREME VAŽE ZA PRAVNA I FIZIČKA LICA (OD 01. 02. '92 PLAĆA SE POREZ 3%)

INSTITUT ZA NUKLEARNE NAUKE "BORIS KIDRIČ", VINČA
CENTAR ZA PERMANENTNO OBRAZOVANJE
11000 Beograd, Nemanjina 4/X
Telefon: 011/ 683-390, 682-486, 641-155/ 107,181
Telefax: 011/ 682-486

KNJIGE U OBLASTI PROGRAMIRANJA, PROGRAMSKIH JEZIKA I PRIMENE RAČUNARA

1. AutoCAD (verzija 10.0)
konstruisanje i projektovanje pomoću personalnih računara
Autori: Boris Damjanović i Petar Damjanović
Šesto izdanje 1991 – latinica, 444 strane, format B 5, broširan povez

2. Uvod u C jezik
Autor: Vladan Vujičić
Četvrto izdanje, 1991. – latinica, 317 strana, format B 5, broširan povez

3. Primena programa SYMPHONY na personalnim računarima
Autor: Dragan Pantić
Treće izdanje, 1990 – latinica, 226 strana, format B 5, broširan povez

4. OS/2 - vodič za korisnike
Autor: Zorica Jelić
Prvo izdanje, 1989 – latinica, 253 strane, format B 5, broširan povez

5. VENTURA - računarsko izdavaštvo
Autor: Predrag Davidović
Treće izdanje, izlazi iz štampe u martu 92.

6. FORTRAN 77
standard sa dopunama za personalne računare
Autori: Vlajko Kocić i Zoran Konstantinović
Drugo izdanje, 1990 – latinica, 422 strane, format B 5, broširan povez

7. UNIX - vodič za korisnike
Autor: Zorica Jelić
Treće izdanje 1991 – latinica, 422 strane, format B 5, broširan povez

8. Primena programa FRAMEWORK III na personalnim računarima
Autor: Dragan Pantić
Prvo izdanje, 1990 – latinica, 326 strana, format B 5, broširan povez

9. Programski alati u matematici
MathCAD, GRAPHER, EUREKA
Autor: Ante Čurlin
Prvo izdanje 1990. latinica, 402 strane, format B-5, broširan povez

10. Primena programa QUATTRO na personalnim računarima
Autor: Dragan Pantić
Prvo izdanje, 1992. latinica, 296 strana, format B-5, broširan povez

11. DOS UKRATKO - verzije 3.3 i 5.0
Autor: Dragan Pantić
Drugo izdanje, 1992. – latinica, 120 strana, format B 5, broširan povez

12. Vodič za VAX/VMS
Autori: Tamaš Kerepeš, Zvonko Oršolić, Saša Matijević
Prvo izdanje 1990 latinica, 512 strana format B 5, broširan povez

13. Primena programa EXCEL na personalnim računarima
Autor: Dragan Pantić
Prvo izdanje, 1991 – latinica, 272 strane, format B 5, broširan povez

14. UNIX - vodič za programere
Autor: Zorica Jelić
Prvo izdanje, 1991 – latinica, 326 strana, format B 5, broširan povez

15. WINDOWS 3.0
Autor: Dragan Pantić
Prvo izdanje 1991 latinica, 273 strane, format B 5, broširan povez

16. PRIMAVERA
upravljanje projektima uz pomoć računara
Autori: Jaroslav Urošević i Jelica Drašković Ostojić
Prva izdanje, 1991 – latinica, 365 strana, format B 5, broširan povez

17. dBASE III+ priručnik
Autor: Milorad Filipović
Prvo izdanje, 1991 latinica, 249 strana, format B 5, broširan povez

18. Osnovi informaciologije i informacione tehnologije
Autor: Ljubomir Dnlović
Prvo izdanje, 1991 latinica 538 strana, format B 5, broširan povez

19. LOTUS 1-2-3 (verzije 3.0 i 3.1)
Autor: Dragan Pantić
Prvo izdanje 1991 latinica, 431 strana, format B 5, broširan povez

20. dBASE IV priručnik
Autor: Ljubomir Lazić
Prvo izdanje 1991 latinica 306 strana, format B 5, broširan povez

21. WORDPERFECT (verzija 5.1)
Autor: Dragan Pantić
Prvo izdanje, 1991 – latinica, 300 strana, format B 5, broširan povez

22. Programiranje u CLIPPER-u 5.01
Autor: Alempije Veljović
Prvo izdanje, 1992. – latinica, 465 strana, format B 5, broširan povez

23. FoxPro
Autor: Dušan Čašić
Prvo izdanje 1992. latinica, 390 strana, format B 5, broširan povez

24. Uvod u strukture podataka
Autor: Miroslav Jocković
Prvo izdanje izlazi iz štampe u januaru '92.

25. ORACLE (verzija 5)
arhitektura i administracija
Autor: Vladimir Milojković
Prvo izdanje izlazi iz štampe u januaru '92.

RAČUNARI, februar 1992

Poručujem (ispod rednog broja knjige upisati broj poručenih primeraka)

Redni broj knige	1	2.	3.	4	5.	6	7	8.	9.	10	11.	12.	13.	14.	15	16	17	18	19	20	21.	22	23	24	25
Broj poručenih primeraka																									

IME I PREZIME
(Naziv preduzeća)
ULICA I BROJ
BROJ POŠTE i MESTO **TELEFON**

Porudzbenicu poslati sa petim primerkom uplatnice (informacije o cenama svakog radnog dana od 8 do 20 satr) na adresu
Institut za nuklearne nauke 'Boris Kidrič", Vinča
Centar za permanentno obrazovanje, Beograd, Nemanjina 4/X.
Uplata se vrši na žiro račun:
INSTITUT ZA HEMIJSKU DINAMIKU i PERMANENTNO OBRAZOVANJE, broj: 60803-603-17361
Troškove poraza na promet i isporuke knjiga snosi poruciiac i nomrruje ih unapred zajedno so uplatom za knjrge



JEZIK ČETVRTE GENERACIJE I RELACIONA BAZA PODATAKA

UOBIČAJENA OGRANIČENJA NE VAŽE

kada izaberete PROGRESS za izgradnju i primenu aplikacija. PROGRESS otvara najširu paletu izbora u odnosu na bilo koji 4GL ili bazu podataka.

PROGRESS vam pruža:

Razvoj. Kompletne OLTP aplikacije se mogu razviti u našem 4GL, mogu se kombinovati SQL i 4GL u okviru procedura ili pozivati 3GL rutine. Pored ovoga, front-end alati ubrzavaju izradu prototipova aplikacija i izbor korisničkih izveštaja.

Performanse. Višeznačna klijent server arhitektura, dozvoljavajuci više servera po bazi podataka, eliminiše zastoje kod čekanja na obradu. Optimalne performanse su obezbedene za stotine korisnika i velike, distribuirane baze.

Baza podataka. Performansama se ne žrtvuje integritet baze podataka. Ugradena kontrola istovremenih aktivnosti sa potpunom fizičkom i logičkom podrškom transakcija osigurava automatsko očuvanje višekorisničkih aplikacija kod 'padanja' sistema.

Prenosivost. Aplikacije se mogu prenositi na stotine računarskih platformi bez menjanja ijednog reda originalnog koda. PROGRESS i vaše aplikacije rade na velikom broju operativnih sistema (DOS, XENIX, UNIX, ULTRIX, VMS, ...), mrežnih protokola i X Window okruženja. Pristupajte podacima na bilo kojoj kombinaciji PROGRESS, ORACLE i DEC baza podataka.

PROGRESS SOFTWARE

Distributer za Jugoslaviju

ABV d.o.o.

Andrićev venac 4, 11000 Beograd
Tel (011) 345-277, 334 990, Fax (011) 345-277

Zašto niste čuli za nas?

Zato što smo se prvo dobro pripremili pre izlazke u javnost se našim programom - *nepunili smo skladište.*

Zato što smatramo da treba biti bolji pre početka, a ne da se tokom rada uhodavamo na vama - *imamo kvalitatne stručnjake.*

Zato što smo smatrali da treba da ponudimo rokove isporuke, bar za većinu popularnijih modela, od najviše 48 sati - *uvek!*

Zato što smo hteli da ponudimo celovito poslovno rešenje, a ne samo računare - *mreže, telefaksove, kurseve za obuku.*

Zato što smo hteli da popuste na količinsku kupovinu prenesemo na Vas - *naše kupce.*

Jesmo li uspeli?

Procenite sami - ovo možete dobiti od nas odmah nakon uplate:

Intersoft 286-16 - *bazni model namenjen kancelarijame i onima sa manjim zahtevima*

Intersoft 386-33 - *model za sve one kojima treba jača platforma za ozbiljniji profesionalni rad, ili kao server u manjim mrežama*

Intersoft 486-33 - *računar za one sa najvećim zahtevima. Namenjen profesionalcima u oblasti CAD/CAM-a, za veoma složene poslove preloma teksta, grafički dizajn i kao server u većim mrežama.*

INTERSOFT

Vaš verni saradnik

Beograd, 29. novembra 43/I, Tel. (011) 325-393, 323-539, Fax. (011) 324-120

COMTRAD

VOLKS-AT 286/12

CONFIGURATION

- 80286 INTEL CPU & TI CHIPSET
- 12 MHZ SPEED
- LANDMARK 15.6
- 1 MB RAM INCLUDED
- 2 SERIAL PORTS
- 1 PARALLEL PORT
- 1 GAME I/O PORT
- SOCKET FOR 80287 COPOCESSOR
- FLOPPY DISC DRIVE 5.25" 1.2 MB
- FLOPPY DISC DRIVE 3.5 " 1.44 MB
- 40 MBYTE AT-BUS HARDDISC
- VGA DISPLAY ADAPTER
- 14" VGA COLOR MONITOR
- MF II KEYBOARD, 102 KEYS
- SERIAL MOUSE 2 BUTTOM

1.991.- DM

ISPORUKA ODMAH!



Build for your satisfaction

386SX (ista konfiguracija) 2.499 DM

SEIKOSHA SP 2000 Ai	**SEIKOSHA SL 92A**
9 iglica/192 cps	**24 iglice/240 cps**
DM 361	**DM 595**

COUPON

Please send me information about:

- 386 SX
- 386 CACHE 33
- 486 ISA
- 486 EISA 33

ComTrad
HIGH SPEED PC S

COMTRAD WORLDWIDE SUPPORT

UNITED STATES • GERMANY • VENEZUELA

YUGOSLAVIA • NETHERLANOS • SPAIN

ComTrad

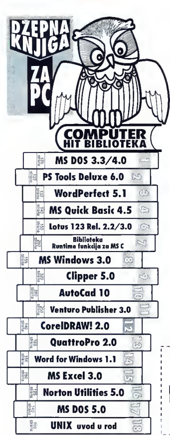
Central European Office Wahlerstrasse 12 Wahler Park 4000 Dusseldorf 30 West Germany
Tel 99-49 211-65-68-020 Fax 99-49 211 65-08-40

Comtrad — Yugoslavia
Vladimira Popovica 6
11000 Beograd
Tel 011-2224-151
Tel 011-2222-652
Tel 011-2224-433/8265
Fax 011-2224-139

Infotrade
Hekurudha 7
38000 Priština
Tel 038-25-630
Fax 038-25-822

Duga Computer
Kumodraška 172
11000 Beograd
Tel 011-23-71-531
Fax 011-23-71-531

Comtrad Kragujevac
Maršala Tita 102
34000 Kragujevac
Tel 034-64-270
Tel 034-60-336
Fax 034-67-117



Džepna knjiga za PC u izdanju Computer hit biblioteke

Trikovi i savjeti iz pera iskusnih korisnika o najpopularnijim programima za personalne računare. Kratke i precizne informacije omogućiće vam da vrlo brzo ovladate korištenjem potrebnog softvera i svu svoju energiju usmjerite na produktivni rad sa računarom.

Obratite pažnju na neka od naših izdanja!

Windows serija!

8. MS Windows 3.0 — Autor: Aco Kordić

Iskoristite prednosti grafičke radne okoline, jedinstvenog korisničkog interfejsa, dinamičke razmjene podataka između različitih aplikacija i potpunog korištenja svih resursa računara.

12. CorelDRAW! 2.0 — Autor: Saša Svitlica

Upoznajte se sa fascinantnim karakteristikama najboljeg programa za crtanje i dizajniranje na PC računarima. U novoj verziji donosi rad u 3D perspektivu, ovojnice, nove fontove, mogućnost kreiranja sopstvenih, konverziju fontova, novi trace program. Oduševljeni korisnici programa Corel DRAW! kažu da Windows i postoje samo zbog njega.

14. Word for Windows — Autor: Amela Abadžić

Upoznajte tekst procesor nove generacije. Iskoristite prednosti grafičke radne okoline, WYSIWYG prikaza i štampanja, dinamičke razmjene podataka sa ostalim Windows aplikacijama, moćnih makroa.

15. MS Excel 3.0 — Autori: A. Abadžić, A. Agović

Trenutno najmoćniji program za tabelarne proračune. Jedini pravi izbor za korisnike koji traže snagu, ali i fleksibilnost i lakoću upotrebe. Pored osnova rada, komandi, funkcija, makroa programa, dat je i prikaz pomoćnih alata: Solver i Q + E.

Nova izdanja iz edicije Džepna knjiga za PC

16. Norton Utilities 5.0 — Autor: Samir Dobrić

Priručnik za rad sa programom koji je vremenom postao standard u oblasti pomoćnih programa namijenjenih svakodnevnom održavanju računara.

17. MS DOS 5.0 — Autor: Samir Dobrić

Uprkos crnim prognozama da će vrlo brzo biti prevaziđen i pasti u zaborav, MS DOS se ove godine pojavljuje u novom ruhu i potvrđuje svoju vitalnost i aktuelnost.

U pripremi

18. UNIX - uvod u rad — Autor: Vladimir Požek

Uvodna znanja za korisnike UNIX-a od mikro do mainframe računara. Knjiga će vas upoznati sa osnovama ovog moćnog operativnog sistema i osposobiti za samostalan rad. Sve komande su potkrijepljene primjerima.

19. Personalni računari grada i primjena — Autor: Darko Grundler

Sve što ste željeli znati o personalnim računarima naći ćete u ovoj knjizi. Na jednostavan i lako čitljiv, a istovremeno stručan i poučan način opisani su građa i primjena personalnih računara. Ovo je knjiga koja će vam omogućiti da sebi olakšate snalaženje u svijetu personalnih računara i uz malo truda sistematizujete svoje znanje.

Molimo vas da nas prije narudžbe kontaktirate u vezi cijena.

Knjige možete naručiti poštom priloženom narudžbenicom ili telefonom na brojeve: 071/270 213, 621-025, 276-680, fax: 071/276-680 od 08 do 17 sati.

U cijene knjiga nisu uračunati troškovi isporuke koji trenutno iznose oko 100.00 dinara po pošiljci.

COMPUTER HIT BIBLIOTEKA
71000 Sarajevo, Sime Milutinovića 4/II

NARUDŽBENICA

NARUČUJEM SLIJEDEĆE NASLOVE IZ EDICIJE "DŽEPNA KNJIGA ZA PC"

1	2	3	4	6	7	8	9	10	11	12	13	14	15	16	17	18

Ime i prezime (naziv firme) ______
Poštanski broj i mjesto ______
Ulica i broj ______
Telefon, telefax ______



DELATNOST : Projektovanje i izgradnja informacionih sistema

- PRODAJA NOVELOVIH PROIZVODA
- PROJEKTOVANJE informacionih sistema
- PROIZVODNJA kompletnog hardware-a (računarske opreme)
- PROJEKTOVANJE I IZGRADNJA računarskih mreža i komunikacija
- IZRADA software-a (aplikativnih programa)
- ODRŽAVANJE I SERVISIRANJE hardware-a i software-a
- NABAVKA repromaterijala (diskete, riboni, perforirani papir)
- OBUKA korisnika

mikronova

NOVELL® AUTHORISED DEALER

HARDWARE



Proizvodnja i isporuka najsavremenije računarske opreme, o konfiguracijama prema zahtevu korisnika:

- **PC/AT-286 kompatibilni računari**
- **PC - 386 kompatibilni računari**
- **PC - 486 kompatibilni računari**

Hard diskovi, flopi diskovi, grafičke karte, monitori visoke rezolucije, fax kartice i modemi.

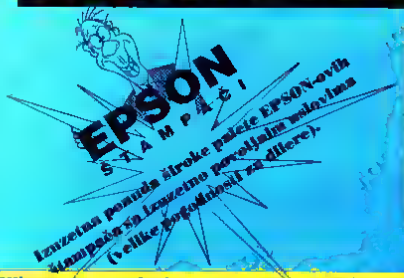
- NETWARE Ver 2.2
- NETWARE Ver 3.11
- MREŽNE KARTICE
- MREŽNA OPREMA

NOVELL RAČUNARSKE MREŽE

SOFTWARE

Proizvodnja optimalno koncipovanog aplikativnog software-a za IBM® PC/AT kompatibilne računare:

- **MATERIJALNO I FINANSIJSKO POSLOVANJE**
- **KADROVSKA EVIDENCIJA, LIČNI DOHOTCI I KREDITI**
- **UPRAVE ORUŠTVENIH PRIHODA**
- **KLINIKE I BOLNIČKA POSLOVANJA,**
- **GINEKOLOŠKO AKUŠERSKE KLINIKE**
- **STAMBENA PREDUZEĆA**
- **TRGOVINSKA PREDUZEĆA**
- **OBRAČUN KAMATA**
- **BUDŽETSKO KNJIGOVODSTVO**
- **PROGRAMI PO ZAHTEVU KORISNIKA**



Izuzetna ponuda širokog pojasa EPSON-ovih štampača za izuzetno povoljnim uslovima (velike pogodnosti za dilere).

Mikronova organizuje za Vas MIKRONOVIN školski centar. U okviru centra održaće se sledeći kursevi:

- **Uvod u operativni sistem MS-DOS**
- **Uvod u CLIPPER**
- **Uvod u GW-BASIC**
- **Uvod u računarske komunikacije i PC LAN-ove**
- **Lokalne računarske mreže (PC LAN) i uvod u mrežni O.S. NOVELL Netware**
- **NOVELL: Kurs za dilere, prodavce i potencijalne kupce, kao i mnogi drugi kursevi.**

Mikronovin školski centar počinje sa radom 24. Februara 1992 godine. Nastava će se održavati u savremenoj opremljenim učionicama (uz prisustvo računara) u ulici Đure Jakšića 3. Obezbedili smo i svu potrebnu literaturu za uspešno odvijanje kurseva. Za sve neophodne informacije obratite se komercijalnoj službi Mikronove.

POSEBNI POPUSTI ZA DALJU PRODAJU,
DILERI DOBRODOŠLI !



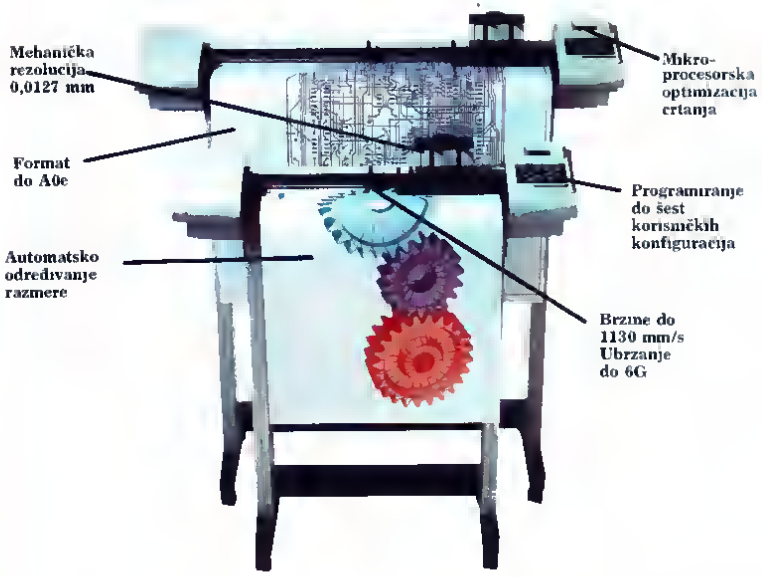
KONTAKT ADRESA :
P.P.mikronova D.O.O. Novi Sad, Lenjinov trg 5/II
Tel.021/52-999 Fax.021/28-042

COMPUTER SHOP Partizanskih baza 21, Tel.021/300-733 Radno vreme 13^{00}-21^{00}

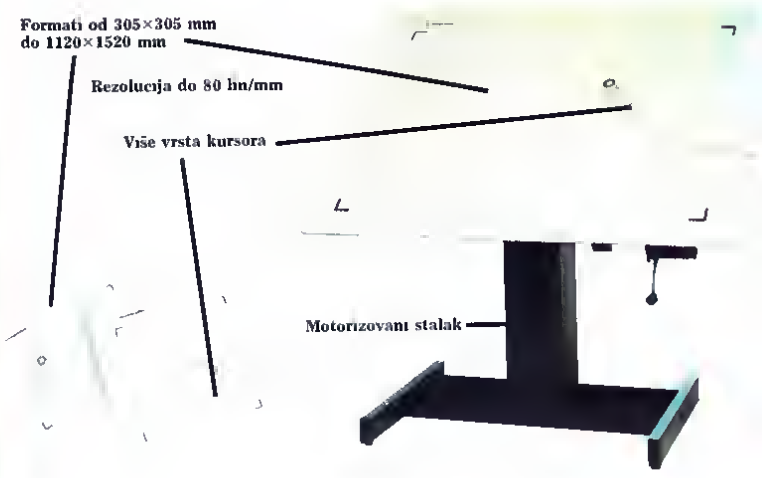
HOUSTON INSTRUMENT

Summagraphics™

Ploteri



Digitalizatori



Skeneri

Sekači

Vrhunske grafičke periferije, softver i inženjering

NOVO: NOVO: NOVO: FAMILIJA PLOTERA EDMP 160 sa još boljim performansama

Ovlašćene prodajne organizacije

SoftWell d o o
Kompjuterski inženjering
Pozorišni trg 1
21000 Novi Sad
Tel 021/51 999 52-396

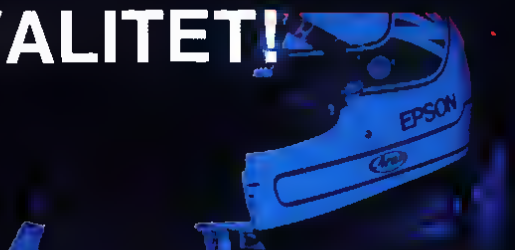
Lanaco
Braća Pavlića H
78000 Banjaluka
Tel 078/39 088 39 162
Fax 33 992

Makpetrol kompjuteri
Kompjuterski inženjering
Ul „Orce Nikolov" bb
91000 Skopje
Tel 091/229 676 221-040
Fax 229-676

kma D.O.O.
11070 N Beograd
Milentija Popovica 9 (Sava centar)
Tel 011/222-19-28
Tel 011/439-618
Tel 011/435-715
Tel 011/222-43-22, lokal 453
Fax 011/455-785, 605-578

EPSON

VELIKO IME, POUZDAN KVALITET!



ÉPSON



COMPUTERS

Beograd, Dalmatinska 76, tel. 011/454-612, fax 011/450-471

EPSON-ove printere prodaju preko ovlašćenih i

INTERC ACOFF Beograd

ASYS COMMERCE Novi Sad

MIKROINOVA Novi Sad

INFOTRADE Pančevo

AUTODESK

AutoCAO R.11
AEC
AutoSHADE
AutoSKETCH
ANIMATOR
3D STUOIO

RASTEREX Graphics solutions

PROFESIONALNE
GRAFIČKE KARTICE·
LIGHT AT,LIBERTY
AutoIcon
RxImage
SpotLight

océ Graphics

PLOTERI SA PERIMA
TERMALNI PLOTERI
GRAFIČKE TABLE
OIGITAJZERI
KOLOR TERMALNI
PRINTERI

Genoa

OSNOVNE PLOČE:
386-33 MHz/64k,
486-33 MHz/256k
GRAFIČKE KARTICE



Chicony

TASTATURE

LAPTOP RAČUNARI

PROJEKTOVANJE SISTEMA
PC RAČUNARI SVIH KONFIGURACIJA
NOVELL MREZE
AutoCAD I POSLOVNE APLIKACIJE

RAČUNARSKI INŽENJERING
BEOGRAD

RACUNARSKI INŽENJERING
CDMPUTER ENGINEERING

Sedište/Head office
Mice Stojkovica 8,
11130 Beograd

Predstavništvo/Representative office
Šumadijski trg 6a,
11030 Beograd,

tel. 38 11 557-592
tax. 38 11 559-845

ŠTAMPAČI

Matrični
- DL 900
- DL 1100
- DL 1200
- DL 3600
- DL 5800

Laserski
- VM 800
- RX 7100 PS
- VM 2200

Linijski
- M 3041
- M 3042
- M 3043

PLOTERI
- FW 3000

Skeneri
- M 3191
- M 3093
- M 3096

Ink-Jet
- Breeze 200

DISKOVI
- M 2611 T/S 45 MB 3 5"
- M 2612 T/S 90 MB 3 5"
- M 2616 T/S 105 MB 3 5"
- M 2613 T/S 130 MB 3 5"
- M 2614 T/S 180 MB 3 5"
- M 2622 T/S 330 MB 3 5"
- M 2623 T/S 425 MB 3 5"
- M 2624 T/S 520 MB 3 5"

- M 2261 T/S 415 MB 5 25"
- M 2263 T/S 780 MB 5 25"
- M 2266 T/S 1266 MB 5 25"
- M 2257 T/S 2 GB 5.25"

- M 2382 1 GB 8"
- M 2392 2 GB 8"

FUJITSU



digit® RAČUNARSKI INŽENJERING

Gradski park br. 2, Hala "PINKI" 11080 Zemun tel. 011 194 442 109 761 fax 612 784



Nothing Is Faster Than A Great Team.

In sailing, great teamwork wins races. The combination of a powerful sailboat and a skilled crew can't be beat. For high performance computing, there's another team that can't be beat: your workstations and our memory.

You see, Dataram memory is the perfect teammate for your workstation. No other memory gives you more speed and high performance than our technically superior boards.

But we give you more than speed. We assure our memory's quality and reliability with a Lifetime Guarantee. We price our memory quite competitively. And we support it with an expert technical staff. In fact, since 1967, we've offered a full line of high performance memory products — with speeds up to 250 MHz.

For the ultimate in high performance computing, team your workstation with our memory. It's an unbeatable combination.

DEC	SUN	HP/Apollo	DG
VAXstation 3100	3/260 3/280	DN2500	AViiON
VAXstation 3200 3900	3/470 3/480	9000/340 DN3000	MV series
VAXstation 2000	4/260 4/280	9000/360 DN3500	
DECstation 2100/3100	SPARC 330 SPARC 370	9000/350 DN4000	**IBM**
MicroVAX II/III/IV	SPARC 490	9000/370 DN4500	RISCSystem 6000
VAX 6000 Systems	1MB/4MB SIMMs		

DATARAM

digit® - RAČUNARSKI INŽENJERING

Gradski park br. 2, Hala "PINKI", 11080 Zemun, tel. 011-194-442, 109-781, fax. 612-784

CITIZEN

IMA DOBRU IDEJU



Vrhunski kvalitet

Atraktivna cena

Dve godine garancije

Obaveštenja kod CITIZEN-ovog partnera za Jugoslaviju:

MerkoM

Kruševac Dom sindikata V/5,
Tel 037/28-966

i sledecih distributera:

ŠUMADIJA — N Beograd
Goce Delčeva 36
Tel 011/600-925
Fax 011/600-395

Jugolaboratorija — Beograd
Cara Uroša br 9
Tel 011/634 192
Fax 011/187-286

Jugodata — Beograd
Save Kovačevića br 6
Tel/Fax 011/444-3176, 453-560

BYTE servis — Beograd
Bulevar JNA 131
Tel 011/464-286

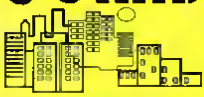


Isporuka SIGURNA

IMTEL U BEOGRADU

garancija 2 godine





MONITOR	MONOHROMATSKI VGA			KOLOR VGA		
HARD DISK / RAČUNAR	40MB	105 MB	210 MB	40MB	105 MB	210MB
486-33, 4MB RAMA, MINITOWER	4510	5180	5880	5030	5830	6530
386-40, 2MB RAMA, MINITOWER	3810	4480	5180	4460	5130	5830
386SX-25,2MB RAMA, STAND. KUĆIŠTE	2680	3350	4050	3330	4000	4700
286-16, 1MB RAMA, STAND. KUĆIŠTE	1860	2550	3250	2530	3200	3900

CENE SU U DM I OBRAČUNAVAJU SE U DINARE PO KURSU MARKE NA SLOBODNOM TRŽIŠTU

ŠTAMPAČI	
HP LASER JET III P	3700
MEM. PROŠ. ZA LJ III P	400
TONER ZA LJ IIP	330
EPSON FX 1050	1450
EPSON LX 800	700

KOPROCESORI.	
AMD 80C287-12	180
80387SX 20	550

MODEMI 2400 BAUDA.	
POCKET	200
EKSTERNI MNP5	400

SOFTVER.	
DR DOS 4.3	100
MS DOS 5.0	250
MS WINDOWS 3.0	350

EPROM PROGRAMER	400
HP-IB KARTICA	550
AD/DA KARTICA	950
STRIMER COLORADO 120MB	800
MIŠ GM-6	55
FLOPI 1,44MB	150

DISKETE. 1.2 MB 1.44MB

NOVELL NETWARE MREŽNI OPERATIVNI SITEM

RAČUNARI SU JAČI KADA SU POVEZANI !

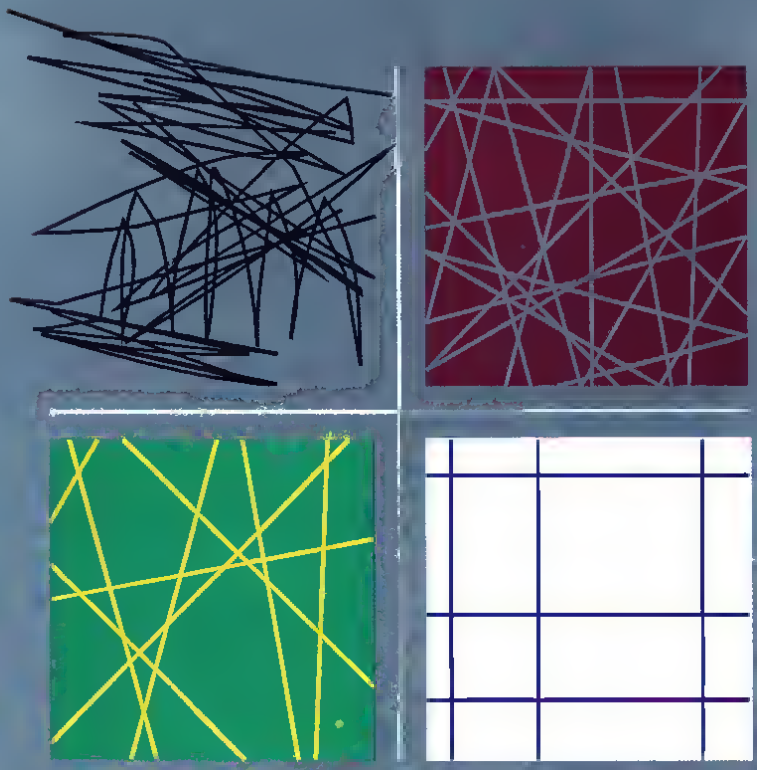


NOVELL NETWARE 386 V3.11 PODRŽAVA DO 100 KORISNIKA	16000 DM
NOVELL NETWARE 386 V3.11 PODRŽAVA DO 20 KORISNIKA	8000 DM
NOVELL NETWARE 286 V2.2 PODRŽAVA DO 50 KORISNIKA	8000 DM
NOVELL NETWARE 286 V2.2 PODRŽAVA DO 10 KORISNIKA	5000 DM
ETHERNET MREŽNA KARTICA NE2000	550 DM

IMTEL - Institut za Mikrotalasnu Tehniku i Elektroniku

Bulevar Lenjina 165 b, 11070 Novi Beograd

Telefon: (011) 134-516, Fax: (011) 138-928, Telex: 12797 IMTELYU,Žiro račun 60815-603-14396



The Art of Networking

Računare treba povezati u mreže da bi se optimalno iskoristili ljudski i mašinski resursi. To je novi korak u razvoju računarstva i informatike. Vreme individualnih i velikih mainframe računara u poslovnom okruženju je prošlo.

Da bi se formirala mreža nije dovoljno samo kupiti kutije. Dođite kod nas sa znanjem o zahtevima koje sistem treba da ispuni u praksi i izradićemo projekat kompletnog informacionog sistema prema Vašim potrebama. Naši stručnjaci su kompetentni konsultanti koji mogu rešiti Vaše probleme u oblasti poslovne obrade. Poslovni softver koji vam nudimo predstavlje vrhunska dostignuća u ovoj oblasti.

Odabrali smo „Novell", najvećeg proizvođača softvera za lokalne računarske mreže, jer se „Most" pouzdaje samo u proizvođeče koji garantuju kvalitet. Sa „Novell" NetWare operativnim sistemima naše mreže ispunjavaju sve Vaše zahteve za povezivanje PC, mini i mainframe računara. Raspolažemo znanjem i iskustvom koje Vam garantuje da će Vaše mreže raditi uz maksimalno iskorišćenje performansi. Pored toga kod nas ćete dobiti kompletnu, stalnu podršku: otklanjanje svih kvarova u radu mreže u roku od 24 časa, kao i telefonsku podršku.

Kupovinom u „Mostu" teko dobijete kompletno rešenje Vašeg problema. Naši sistemi su projektovani tako da budu tehnički fleksibilni i finansijski povoljni. Obratite nam se zahtevom za ponudu za Vaš konkretan problem, ili zahtevajte listu naših referenci.

CENE RAČUNARA

PC AT 286/16MHz	1,790 DEM
PC AT 386/25MHz	3,300 DEM
PC AT 386/33MHz/cache 64kB	3,600 DEM
PC AT 486/33MHz/cache 256kB	6,150 DEM

OSNOVNE KONFIGURACIJE

Stono kućište, 1MB RAM memorije, tvrdi disk 52MB IDE, floppy disk 5.25"/1.2MB ili 3.5"/1.44MB, »Hercules« grafički adapter, monohromatski monitor, tastatura, MS DOS 4.01, uputstvo na srpskom

DOPLATE ZA VEĆE ZAHTEVE

VGA kolor monitor »Samtron«, SVGA adapter 1024x768/512KB »Trident« TVGA 8900C	600 DEM
Tvrdi disk 130MB IDE	550 DEM
Mini tower kućište	120 DEM
»Genius« trackball	170 DEM
»Genius« ručni skener	340 DEM
»SONY« CD-ROM CDU-510-12	990 DEM

UZ KOLOR RAČUNARE: MIŠ I WINDOWS 3.0 NA POKLON!

CENE ŠTAMPAČA

»EPSON« LX-400, 9-pinski, A4	575 DEM
»EPSON« LQ-200, 24-pinski, A4	890 DEM
»EPSON« LQ-570, 24-pinski, A4	1,250 DEM
»EPSON« FX-1050, 9-pinski, A3	1,250 DEM
»Hewlett Packard« HP LaserJet IIIP, laserski, 1MB	3,450 DEM
»Hewlett Packard« HP LaserJet III, laserski, 1MB	4,800 DEM

Garancija 1 godina • Isporuka odmah • Dinarsko plaćanje po dnevnom kursu DEM

»MOST« računari • »Hewlett Packard« laserski štampači • »EPSON« štampači • »Novell « računarske mreže • »Microsoft« operativni sistemi • Poslovni i grafički sistemi

BRUG Computers BRUCKE Computers MOST Computers

MOST, Beograd, Kralja Bodina 2, tel. 011/422-170, fax. 011/426-144

ZAŠTO BI VAŠ NOVI RAČUNAR TREBALO DA BUDE MOST





COMPAQ

U TOKU FEBRUARA MESECA
IZUZETNO POVOLJNE CENE
LAPTOP RAČUNARA
COMPAQ SLT 286M40

COMPAQ računari su odgovor
na stvarne potrebe korisnika

Ovlašćeni distributer

MS MICROSYS

21000 Novi Sad, Temerinska 49
Tel: (021) 611-366, fax: (021) 20-544

Poslovnica Beograd, Dobračina 39
Tel/fax: (011) 622-581

Jednostavno, mi smo pouzdani

Jedan za sve

O R A C L E razvojno okruženje i sistem za upravljanje relacionim bazama podataka omogućuje razvoj aplikacija za sve popularne računare i operativne sisteme.

Aplikacije jednom razvijene u ORACLE okruženju mogu se bez ikakvih izmena prenositi na bilo koji sistem - od samostalnog PC računara preko „klijent-server" konfiguracije na LAN u minikompjutera sve do najvećeg IBM računara.

Aplikacije koje koriste ORACLE alate i relacioni sistem za upravljanje bazama podataka mogu se izvršavati na oko 80 različitih platformi među kojima su: MS-DOS, Macintosh, OS2, UNIX (AT&T, SCO, ISC, Motorola, HP, Convergent, Unisys, NCR, Prime, Sequent, Bull, Convex, Sun, Apollo, Olivetti, Alcatel, Phillips, Arix, Siemens, Nixdorf, Pyramid, Next, Tektronix, DEC, MIPS, ICL, IBM, ...), Bull (GCOS7, HVS), VAX/VMS, IBM (MVS, VM, DOS/VSE), ICL, VME, Siemens BS, Stratus VOS, Tandem Guardian, DG AOS/VS, Wang VS, Prime PRIMOS, CDC NOS/VE, HP MPE/XL.

Sve aplikacije mogu raditi na istom sistemu na kome se nalaze i podaci ili mogu koristiti sve popularniji način rada „klijent-server". Tada se korisničke aplikacije izvršavaju na radnim stanicama (one mogu biti i personalni računari) vezanim na jedan ili više snažnijih računara gde se nalaze podaci pod kontrolom sistema za upravljanje bazama podataka.

ORACLE alati mogu da koriste podatke distribuirane na više računara u mreži na kojima radi ORACLE ili proizvodi drugih firmi (kao što su IBM DB2 i SQL/DS i DEC RMS).

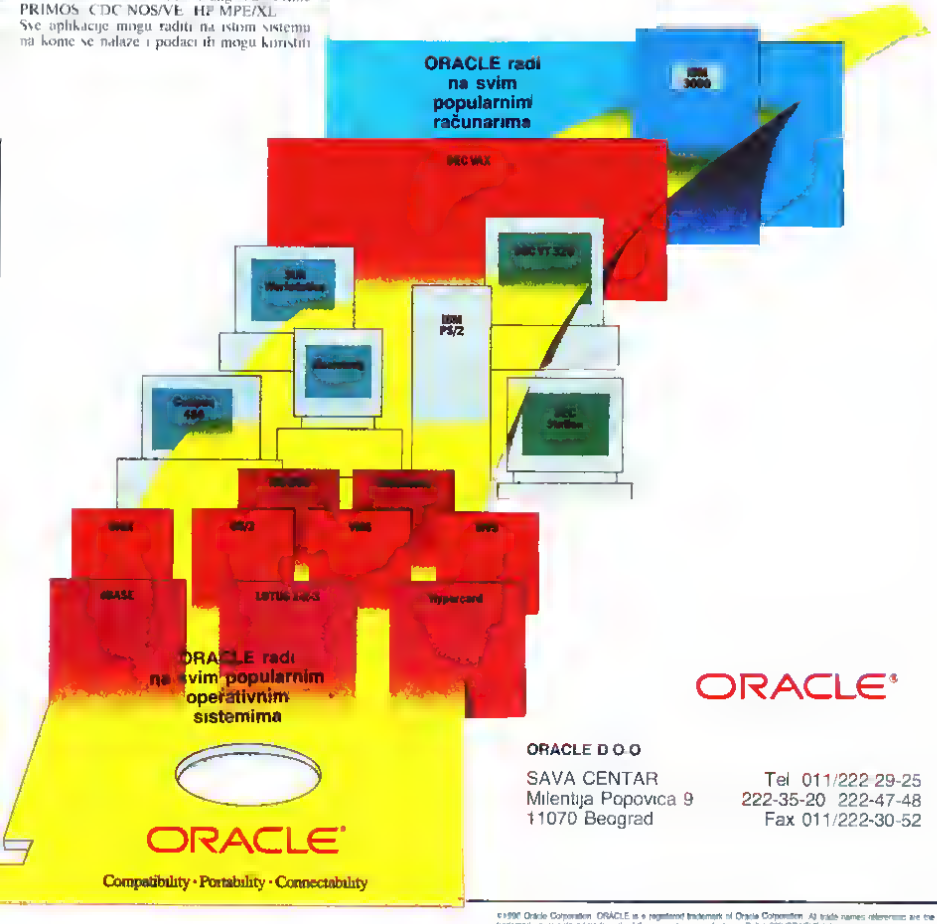
ORACLE integrisani paket alata za razvoj aplikacija uključuje moćan jezik četvrte generacije PL/SQL, generator transakcionih interaktivnih aplikacija SQL*Forms, generator izveštaja SQL*ReportWriter, interaktivnu grafiku ORACLE*Graphics, elektronsku poštu ORACLE*Mail i mnoge druge komponente.

Razvoj aplikacija podržan je potpuno integrisanim familijom proizvoda za projektovanje poslovnih sistema pomoću računara: CASE*Designer, CASE*Dictionary, CASE*Generator i CASE*Project.

Oracle je najveći proizvođač softvera za upravljanje bazama podataka i alata za razvoj interaktivnih aplikacija.

Oracle obezbeđuje tehničku podršku i školovanje za svoje proizvode. Oracle ima najveću organizaciju za usluge i tehničku podršku među nezavisnim softverskim firmama.

Oracle ima više od 100 korisnika na teritoriji Jugoslavije (sa računajući instalacije na personalnim računarima).



ORACLE

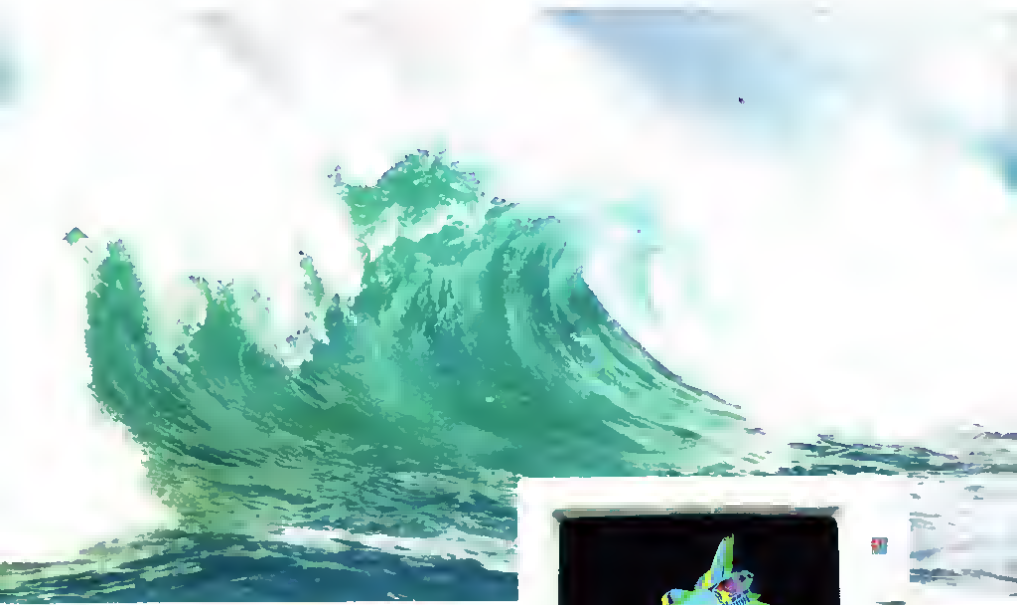
ORACLE D.O.O.

SAVA CENTAR
Milentija Popovica 9
11070 Beograd

Tel 011/222-29-25
222-35-20 222-47-48
Fax 011/222-30-52

©1990 Oracle Corporation. ORACLE is a registered trademark of Oracle Corporation. All trade names referenced are the service mark, trademark, or registered trademark of the respective manufacturer. Call 1-800-ORACLE 1 for hardware and software requirements.

DAEWOO





Računari vrhunskog kvaliteta od popularnog 286 do najmoćnijih 486

Sa tradicijom koja počinje od same pojave personalnih računara

Izradeni uz pomoc vrhunske tehnologije, s pažnjom i beskompromisnim kvalitetom

Sa podrškom koja pokazuje naše poštovanje prema korisniku

Sa cenom koja omogucava da se sutrašnje tehnologije koriste vec danas

Javite nam se, ili posetite naš demo centar

kmaD D.O.O.

Posebne promotivne cene
Dinarsko plaćanje. Isporuka odmah.

11070 N. Beograd
Milentija Popovica 9
(Sava centar)

Tel. 011/222-19-26
Tel. 011/439-618
Tel. 011/435-715

Tel. 011/222-43-22, lokal 453
Fax. 011/455-785, 605-578

Nenad Batoćanin

BAZE PODATAKA Clipper 5.01

PRO ET CONTRA
CLIPPER: SISTEM ZA RAZVOJ BAZA PODATAKA
KAKO NAUČITI CLIPPER
CLIPPER PROGRAMSKI JEZIK
OD PRG DO EXE DATOTEKE
OSNOVNE NAREDBE
BROJEVI
STRINGOVI
DATUMI
LOGIČKI TIP I OPERATORI POREĐENJA
TIPOVI I MEMORIJSKE KLASE PROMENLJIVIH
NIZOVI
PROSLEĐIVANJE PARAMETARA
PROŠIRENI SKUP NAREDBI
KONTROLA EKRANA
UČITAVANJE VREDNOSTI I MENIJI
BOJE
KONTROLA TASTATURE
KODNI BLOKOVI
OBJEKTI
MAKRO OPERATOR
PRETPROCESOR
BAZE PODATAKA
BAZA PODATAKA
BAZA PODATAKA I CLIPPER
ALIASI
PREGLED DATOTEKA
KREIRANJE DATOTEKA
AZURIRANJE DATOTEKA BAZE PODATAKA
PRETRAŽIVANJE I IZDVAJANJE
INDEKSIRANJE
VIŠESTRUKE RELACIJE
RAZVOJ PROGKAMA U CLIPPER-U
DBU JEDNOSTAVNE OPERACIJE SA DBF DATOTEKAMA
CLIPPER PREVODILAC
RTLINK
DEBUGGER
RMAKE
RUN TIME OKOLINA
DINAMIČKI OVERLEJI

...nar

PRO ET CONTRA

Kada se pomene *Clipper* niko ne ostaje ravnodušan. Zaista, taj programski jezik (ili šta je već) neobično jasno deli programera u dva potpuno suprotstavljena tabora. Rezultate ovakve podele možete videti u slučnoj štampi pa čak i u "Računarima". Nevažno ni jedna od ovih grupa nikad neće priznati da je ona druga u pravu, pa ma šta se desilo. Zato rasprava ima svrhe samo kao orijentacija onoj trećoj grupi – ljudima koji se spremaju da rade neki posao po prvi put ali ne znaju za koji programski jezik da se opredele.

Kako se odlučiti? Računarska štampa vrvi od raznih "graničnih" napisa na osnovu kojih vrlo teško možete zaključiti da li će baš to biti najbolje za vas. U "Računarima" ste mogli da pročitate razne stvari o *Clipper*u ali ste takođe mogli da pročitate i dobre prikaze konkurentskih proizvoda, što će svakako doprineti da donesete bolju odluku. Moram da primetim da je biti protivnik *Clipper*a trenutno moderno, pa se mogu naći i potpuno neuka (i netačne) tumačenja. Koja četu i sami uz mlin malan trud prepoznati. Ipak, ovakva tumačenja su najopasnija jer ponekad stvaraju iluziju da su to neki opšteprihvaćeni stavovi.

Clipper je najkraće rečeno programski jezik za razvoj baze podataka. Argumenata za njegovo korišćenje ima puno, ali presudnih iskreno nema previše, uglavnom odlučuju neki sporedni efekti poput vašeg prijatelja u čiji pravilan izbor imate poverenje. Ipak ne kažem da treba imati u vidu.

Prvo *Clipper* je nastao kao jedan od mnogih prevodilaca za *dBASE*. Bez obzira što je najnovija verzija (a i prethodne u dobroj meri) izrasla u samostalan proizvod izvesno "zakrpe" su ostale pošto je programski jezik *dBASE* na nivou sintakse vrlo sumnjivo sklepan. *Clipper* je sjajnim prilikama nasledio vrlo klimavu sintaksu. Nove verzija je to prevazišla, ali je zato postala prilično nekompatibilna sa prethodnom.

Dužina postojanja proizvoda sa druge strane, uliva će povoljno na pouzdanost greške su detaljno izučene i ne postoji opasnost da će vas sutra nešto "sačekati" iz ugla.

Možda je najvažnije da je *Clipper* kakav jeste de facto postao standard. To znači da novi proizvodi moraju na neki način da se upoređuju sa njim, pa makar mu bili i direktna konkurencija. Mnoge softverske firme su za njega izdale gomilu dodatnih (ili počav od raznih biblioteka funkcija za proširenje jezika, pa do raznih generatora programa i izveštaja... tu vam skoro sigurno ništa neće nedostajati). U Jugoslaviji je *Clipper* još popularniji, skoro da ne postoji ozbiljniji program za poslovnu obradu podataka koji nije napisan ne *Clipper* u.

Vrlo često zamerke upućene na adresu sporosti i neefikasnosti *Clipper*-a jednostavno nisu tačne. Za dokaz pogledajte bilo koji *Clipper* program, čak ga je pisao i iole iskusniji programer, nigde nećete uočiti ni traga sporosti.

Na kraju, ne treba zanemariti vrlo značajnu pojavu verzije 5.01 koja je nešto sasvim novo: nikad moćnije, potpuno novi koncepti i moćni mehanizmi mogu do krajnjih granica unapredriti vaše programiranje. Ako vam treba pouzdan i veoma moćan sistem za razvoj složene aplikacije, slobodno se opredelite na *Clipper*, ja verujem da će vas ikad izneveriti!

Zaista, šta je to *Clipper*? Pogrešno je mišljenje da je *Clipper* samo programski jezik – to je u stvari kompletan sistem za razvoj programa koji će raditi sa bazama podataka. Konačenje *Clipper*a u druge svrhe je moguće, ali ipak neprimereno – za programe koje na rade sa puno podataka ne treba koristiti *Clipper*. Osim programskog jezika, koji je nastao iz *dBASE* jezika, tu je i čitav niz alatki koje će vam olakšati svakodnevni rad na bazama podataka: dibager, specijalni linker *make* program, *DBU* za jednostavne operacije nad datotekama baze podataka i sl.

Clipper programski jezik nije ni malo standardan – pri tom ne mislim na standardne konstrukcije za tok programa koje veoma liče na sličane u ostalim jezicima već na veoma specifične operacije koje su ugrađene u samu osnovu jezika. To je apsolutno novitet u odnosu na klasične jezike – ako u *C* u pokušaje da radite bazu podataka, moraćete da nabavite (ili sami napišete) procedure za pristup podacima, korisnički interfejs, itd. A *Clipper* **sve** to sadrži u jo u osnovnom jeziku. Verovatno je to i jedan od razloga ogromne popularnosti koju je postigao: sve što vam treba na jednom mestu. Ogroman broj funkcija i naredbi obezbeđuje veoma dobru kontrolu ekrana, tastature, štampača, podataka u bazi podataka, datoteka na nivou DOS-a i uopšte svega što će vam zatrebati (*Clipper* sadrži preko 400 različitih funkcija i naredbi). Da sve bude još lepše, osnovna koncepcija takvog sistema je **otvorenost** za izmene – sve što vam se ne sviđa možete promeniti na krajnje jednostavan i legalan način.

Sasvim je sigurno da Clipper danas predstavlja zaokružen, pouzdan i moćan sistem za razvoj baza podataka čiju je primamljivost teško odoleti. Naravno, Clipper nikako nije savršen proizvod. On ima čak ne verovatno mnogo nedostataka za njegovu reputaciju. Na svu sreću, većina tih nedostataka je više estetske prirode.

Ali i pored velikog broja konkurentskih proizvoda zaista je teško naći proizvod koji može u svim tačkama da konkuriše Clipperu. To je sa jedne strane sasvim razumljivo, jer na strani Clippera je velika tradicija i iskustvo koje je vrlo teško oboriti preko noći.

KAKO RADI CLIPPER

Knjiga o Clipperu ima zaista puno, pa i na našem jeziku. Na žalost, ogromna većina takvih knjiga je praktično neupotrebljiva. Većina autora na me posle kraćeg uvoda prelazi na opširan spisak naredbi i funkcija koji nije baš preporučljiv za učenje jezika, a na kraju daje kao dodatak (koji može da zauzme i polovinu knjige) ogroman listing gotovog programa kako bi "čitalac naučio sve na primeru". Naravno, ovakve knjige je najlakše napisati, ali su i najmanje korisne. Daleko fleksibilniji spisak naredbi i funkcija možete naći u samom Clipperu u obliku rezidentnog programa NG, a veliki listing (obično loše komentarisan) je zaista zadnje sredstvo za učenje.

Ovaj tekst ima za cilj da donekle popuni prazninu u literaturi. Prosečan početnik u Clipperu obično nije početnik u programiranju - zato se neću previše zadržavati na osnovnim stvarima. Clipper ima veliki broj "zakrpa" nastalih zbog kompatibilnosti sa dBASE-om i starijim verzijama samog Clippera. Što je veliki napor trebam terati. Ovde će takve stvari biti jednostavno preskočene, i za vas i za vaše programe je dobro da takve naredbe i funkcije ne koristite. Ceo tekst se odnosi na verziju Clipper a 5.01, razlike u odnosu na prethodne verzije neće biti posebno označavana.

Ovde neće biti govora ni o svim naredbama i funkcijama. Neće čak biti obrađene ni sve mogućnosti spomenutih naredbi. To je zato što je Clipper egroman, detaljan pregled mogućnosti zahteva daleko više prostora nego što zauzima ovaj tekst. To je i glavni problem kod većine knjiga: treba obraditi sve, ali ne treba zatrpati čitaoca podacima. Zato će ovde biti o bradena samo osnovne stvari koje će vam omogućiti da samostalno počnete da pišete programe. Kada vam budu zatrebale dodatne mogućnosti naredbi, lako ćete ih naći pomoću rezidentne baze podataka NG. A ostalo - to je stvar iskustva kome vas niko ne može naučiti.

Pri čitanju ćete možda primetiti izvesnu (namernu) nedoslednost: neki operatori će biti upotrebljeni pre nego što su definisani. Naime, svaki udžbenik za početnike mora da rešava jedan ni malo jednostavan problem: mrežnu strukturu opšte sintakse i semantike jednog programskog jezika predstaviti linearno. Tj. uvek sa ostanjeti samo na ono što je već rečeno. Ovde se koristi to što čitalac nije pravi početnik i verovatno zna neke osnovne stvari (na primer da A == B označava poređenje promenljivih A i B), tako da tekst ponekad "ide" malo unapred.

Naredni tekst je podeljen u četiri dela. U prvom delu se govori o osnovama jezika. Tu možete naći kako se pišu standardne programske konstrukcije, kakvi tipovi i klase promenljivih postoje i slično. Ovaj deo je elementarni uvod u Clipper kao programski jezik - u njemu nećete naći ništa što je specifično za baze podataka. Osim osnovnih naredbi za kontrolu toka tu ćete naći kako se program prevodi i linkuje. Ovaj deo obavezno pročitajte ako ste početnik - on će vam pomoći da počnete sa radom.

Drugi deo sadrži opis Clipper ovih nestandardnih konstrukcija za kontrolu ekrana, tastature, boja, nekih specifičnih tipova podataka i operatora. Pošto je većina obrađenih stvari specifična za Clipper 5.01, ovaj deo teksta može biti interesantan korisnicima starijih verzija.

Treći deo je posvećen delu naredbi i funkcija za rad sa bazom podataka. To je u stvari opis najvažnijeg dela Clippera kao programskog jezika za kreiranje baza podataka.

Treći deo je kratak priručnik za razvoj programa, tu su, između ostalog, opisani razni konani pomoćni programi koji se isporučuju za Clipper, a još uvek se vrlo slabo koriste, što sigurno ne zaslužuju. Opisan je i sistem po kome radi Clipper program, dinamički overlej, virtuelna memorija itd.

CLIPPER PROGRAMSKI JEZIK

Osnovna koncepcija Clipper programskog jezika slična je koncepciji standardnih programskih jezika, kao što su Pascal ili C. To prosto znači da program u Clipperu jeste program napisan u nekom standardnom programskom jeziku. Pogledajte jedan jednostavan primer Clipper programa.

```
PROCEDURE Main
   ? "Zdravo!"
RETURN
```

Glavni program se počinje sa **PROCEDURE Main** a završava sa **RETURN**. Jedina naredba u našem programu (?) služi za ispisivanje teksta na ekran. Kada ovaj program upišemo u datoteku korišćenjem nekog ASCII editora, potrebno je da **prevedemo** (kompajliramo) a zatim i **povežemo** (linkujemo) program. Na primer, datoteka u koju smo upisali program može se zvati **PROBA.PRG**. Ekstenzija **PRG** je uobičajena za Clipper programe i treba je poštovati.

OD PRG DO EXE DATOTEKE

Sada pozivamo prevodilac kucanjem sledeće komande

```
CLIPPER proba /n
```

Posle ovoga se na disku pojavljuje datoteka **PROBA.OBJ**. Da bi se program izvršavao treba još izvršiti linkovanje

```
RTLINK FILE proba,
```

Rezultat rada Clipperovog linkera je izvršni program **PROBA.EXE**. Startovanjem ovog programa na ekranu se ispisuje tekst **Zdravo!** Naizgled, sve teče baš kao u slučaju klasičnog programskog jezika. Međutim, ako pogledate veličinu dobijene datoteke, za trenutak ćete se zaprepastiti: više od 100K! Možda ćete se zapitati - koliki li će biti kod kad se napiše više naredbi. Na sreću, situacija nije baš takva. Clipper i nije namenjen da se na njemu pišu kratki programi, po put našeg primera, dobijena datoteka sadrži ogroman ne skončan kod za rad sa bazama podataka.

Da bi prethodni postupak donekle ubrzali, ne morate posebno kucati poziv kompajlera i linkera. Uz Clipper se dobija jednostavna komandna datoteka **CL.BAT** sledeće sadržine

```
CLIPPER %1
IF NOT ERRORLEVEL 1 RTLINK FI %1,
```

koja omogućava da se istovremeno izvrši i prevođenje i linkovanje jednostavnim **CL proba**. Da bi ovu komandnu datoteku koristili za primere u tekstu, prvu liniju promenite u **CLIPPER %1 /N**. Znači, sada sve primere prevodimo i linkujemo komandom **CL <naziv datoteke>**.

Primetili ste da je pri prevođenju navedena opcija /N. Ona isključuje jednu prilično nezgrapnu osobinu Clipper prevodioca da proceduru u datoteci automatski dodeli ime te datoteke. Tako, ako program pišete u datoteci **PROBA**, podrazumevaće se da se procedura u toj datoteci zove **PROBA** što je po meni izuzetno nezgodno i nepraktično. Ova osobina se provlači od dBASE a i nikako je ne treba koristiti. Znači, opcija /N treba da bude stalni gost u komandnoj liniji za prevođenje.

Na ostale opcije RTLink-a i Clipper prevodioca ponovo ćemo se vratiti u drugom delu ovog teksta.

OSNOVNE NAREDBE

Naredbe za kontrolu toka programa su standardne i sigurno ćete se brzo navići na njihovu sintaksu. U sledećem primeru su navedene neke od osnovnih naredbi:

```
PROC Main
   IF   Pristup()
      ? "ZABRANJEN PRISTUP  !
      RETURN
   ELSE

      ? "PRISTUP DOZVOLJEN .."
   END IF

   Kraj := F
   WHILE ! Kraj
      Izbor := Menu ()
      DO CASE
         CASE Izbor == 0
            Kraj .   T.
         CASE Izbor  = 1
            ? "Prva opcija"
         CASE Izbor    2
            ? "Druga opcija"
         OTHERWISE
            ? "Nepoznata opcija"
      END CASE
   END DO
RETURN
```

Primetite da je umesto reči **PROCEDURE** napisano **PROC**. Clipper dozvoljava skraćivanje četiri kon čenih ključnih reči što može biti zgodno, ali samo uz umerenu upotrebu.

Sledi sasvim standardna **IF** naredba. Posle konstrukcije **IF <Uslov>** navode se naredbe koje se izvršavaju u slučaju da je uslov tačan. **IF** naredba se završava sa **END IF**. **IF** naredba može realizovati višestruko grananje opcijama **ELSEIF** i **ELSE**. Na primer

```
IF x    0
   ? "NULA"
ELSEIF x > 0
   ?  "POZITIVAN"
ELSE
   ? "NEGATIVAN"
END IF
```

Važno je da naredba **RETURN** koja je u prethodnom primeru upotrebljena nema veze sa stvarnim (fizičkim) krajem programa - ona je izvršna naredba, označava kraj rada procedure (u ovom slučaju glavnog programa) i može se upotrebiti bilo gde u programu.

U nastavku programa je **WHILE** petlja: sve dok je naveden logički uslov tačan, izvršavaće se naredbe navedene između **WHILE** i **END DO**. Naredba dodeljivanja se u Clipperu označava sa := Tako naredba **Izbor := Menu()** označava da da se prvo računa vrednost funkcije **Menu** i da se zatim ta vrednost dodeljuje promenljivoj **Izbor**.

Sledi još jedna standardna konstrukcija **CASE** i između rezervisanih konstrukcija **DO CASE** i **END CASE** navode se konstrukcije **CASE <Uslov>** iza kojih slede nizovi naredbi koje će se izvršiti ako je zadati uslov tačan. Uvek se izvršava **tačno** jedna od navedenih grana. Na primer, ako je u našem primeru funkcija **Menu** vratila vrednost 2, na ekranu će se ispisati tekst **Druga opcija**. Uslovi iza **CASE** naredbe ne moraju biti srodni - možete pisati uslove proizvoljne složenosti, pa čak i takve koji će se znati tek u vreme izvršavanja programa. Kako se to radi, videćemo malo kasnije. U koliko ni jedan uslov nije ispunjen, izvršiće se naredbe navedene posle **OTHERWISE**. Ako funkcija vrati vrednost nula, završiće se naredba **Kraj := T**. Umesto ove naredbe moglo je da stoji **EXIT**. Ova naredba označava prekid rada tekućeg ciklusa. Tako će se za naš primer program završiti funkcija **Menu** vrati vrednost 0.

Prethodnim naredbama treba dodati i sasvim standardnu **FOR NEXT** petlju

```
FOR i :  1 TO 108 STEP 8 1
   ? i
NEXT
```

Promenljiva I ne mora biti celobrojna, pa korak može biti i realan broj. Ako je korak jedan (najčešći slučaj) piše se samo

```
FOR i :  1 TO 100
   ? i
NEXT
```

Velika mana Clipper-a je što ne poznaje celobrojne promenljive, tako da se sve radi sa realnim promenljivim, pa su operacije poput **FOR** petlje znatno usporene. Srećna je okolnost to što sa ... ma kako to

čudno izgledalo – **FOR** petlja veoma retko koristi. Nau me nju obično koristimo kad primenjujemo funkciju na neki niz, a to se daleko efikasnije radi korišćenjem tzv. kodnih blokova.

Svaka naredba (odnosno konstrukcija) se piše u jednom redu. Pošto naredbe mogu biti relativno kratke, omogućeno je pisanje više naredbi u jednom redu – tada ih treba međusobno razdvojiti simbolom ";" (tačka zarez). Na primer

```
IF LastKey() == K_ESC ; RETURN ; END IF
```

Sa druge strane, neke naredbe mogu biti veoma duge. To se naročito odnosi na duge nizove opcija u naredbama za rad sa bazom podataka. Kada deo naredbe treba preneti u sledeći red, kraj prvog reda treba označiti isto simbolom ";". Na primer

```
IF LastKey() == Chr (Key)  .AND. Eval
(Cblk, 77) <> 0 .AND. ;
      NextKey() == Chr(K_ENTER)
   ? "[Ok]"
END IF
```

Komentari se mogu označavati na različite načine. Ranije verzije su dozvoljavale oznake "*" i "&&" kao i naredbu **NOTE** za početak komentara. Novi (i preporučljivi) načini su preuzeti iz *C*-a. Linija koja je komentar označava se sa "//"(dve kose crte). Može se komentarisati i blok oznakama "/*" i "*/". Na primer

```
/* PROGRAM ZA PRORACUN PLATA
   ULAZ    POENI
   IZLAZ   PLATA
*/
* Ovo je komentar
// I ovo je komentar
```

BROJEVI

Clipper poznaje nekoliko različitih tipova podataka: **broj**, **string**, **logički tip**, **datum**, **kodni blok** i **objekt**.

Brojevi se koriste uobičajeno i ne pravi se razlika između celih i decimalnih – u stvari, *Clipper* podržava samo realne brojeve tipa **double**. Postoje sve standardne operacije potrebne za rad sa brojevima: +, -, *, /, ** (stepenovanje), % (ostatak pri deljenju) itd. Na primer

```
x := 123 + a
y := (y - 2) / (z % 2)
```

Za brojeve su vezana još dva važna operatora pozajmljena iz *C*-a: ++ i -- operatori. Ovi operatori se mogu primenjivati samo na **promenljive** koje sadrže numeričke vrednosti (kasnije ćemo videti za slično važi i za datume) i povećavaju (smanjuju) njihovu vrednost za **1 za vreme izračunavanja izraza**. Na primer posle sledećih naredbi promenljiva **Y** će imati vrednost 4, ali će kao sporedni efekat **X** dobiti vrednost 5

```
x := 4
Y := x++
```

Ovi operatori se najčešće upotrebljavaju u izrazima, pri čemu se kao sporedni efekat računanja izraza menjaju vrednosti promenljivih. Na primer

```
a := 1
? a++
```

U ovom primeru biće odštampan broj 1, ali će se vrednost promenljive **X** promeniti na 2. Pogledajmo sličan primer

```
a := 1
? ++a
```

Sada će vrednost promenljive **X takođe** biti broj 2, ali odštampaće se broj 2, jer se povećavanje promenljive **X** vrši **pre** računanja izraza. Ovi operatori se mogu upotrebljavati i u mnogo komplikovanijim izrazima. Sasvim slična pravila važe i za operator --.

U programu se vrlo često pojavljuju naredbe poput **X := X+N** ili **X := X-N**. *Clipper* omogućava skraćeno pisanje ovih naredbi

```
X += N
X -= N
```

Kao i ostalih sličnih konstrukcija

```
X *= N    je isto sto i    X := X * N
X /= N    je isto sto i    X := X / N
X **= N   je isto sto i    X := X ** N
```

STRINGOVI

String je u *Clipper*-u proizvoljan niz simbola. Može biti dugačak do 64K, što je sasvim dovoljno za većinu primena. U *Clipper*-u se stringovi koriste vrlo često, zato postoji bogat skup operatora i funkcija koje rade sa stringovima. String se piše kao niz simbola između navodnika ili apostrofa

```
String := "Ovo je string"
specSym := '"@$"&'
```

Bez obzira na ovu slobodu, u programima uvek koristite jedan način obeležavanja. Drugi način služi ako je potrebno definisati string u kome je jedan od simbola baš " - tada se koristi drugi simbol za označavanje početka i kraja stringa. Za stringove su najvažniji operatori spajanja (konkatenacije). Na primer

```
x := "ABC "
y := "DE"
s1 := x + y
s2 := x - y
```

Operator + označava normalno spajanje, a - spajanje bez blanko simbola na kraju prvog operanda. Tako će posle ovih naredbi biti **s1 == "ABC DE"**, a **s2 == "ABCDE"**. Za stringove je vezan jedan važan relacioni operator **$** (podstring). Pomoću njega se ispituje da li je string sadržan u nekom drugom

```
"ABC"  $ "ABCDEF"    --> TACNO
"abC"  $ "ABCD"      --> NETACNO
"&"    $ "a & y"     --> TACNO
```

Veoma važne su i funkcije koje se koriste za rad sa stringovima, slične odgovarajućim *BASIC* funkcijama

```
SubStr (s,p,l) -> Deo stringa S od pozicije P, duzine L.
Left (s, l)    -> Prvih L simbola stringa S
Right (s, l)   -> Zadnjih L simbola stringa S
RTrim (s)      -> String S bez blanko simbola na kraju
LTrim (s)      -> String S bez vodecih blanko simbola
AllTrim (s)    -> String S bez vodecih i krajnjih blanko simbola.
At (s1, s2)    -> Pozicija stringa S1 u stringu S2
Len (s)        -> Duzina (broj simbola) stringa S
```

Veza između celih brojeva i stringova je uobičajena: funkcija **Chr** konvertuje dati ceo broj u odgovarajući simbol po ASCII tabeli, a **Asc** konvertuje prvi simbol u stringu u kod po ASCII tabeli.

Veoma često se koristi funkcija **Replicate** koja multiplicira zadati string proizvoljan broj puta. Na primer, ako je potrebno ispisati liniju sastavljenu od 80 simbola "*", dovoljno je napisati **Replicate ("*", 80)**. Ova funkcija vraća string i može se ravnopravno koristiti u izrazima. Njoj srodna funkcija je **Space** koja vraća prazan string određene dužine

Space(4) > " "

DATUMI

Pošto se pri radu sa bazama podataka vrlo često radi sa datumima, *Clipper* podržava tip podataka **datum**. Sa datumima se vrlo jednostavno i logično radi. Na primer, posle naredbe **d := Date()** promenljiva **D** će dobiti vrednost tekućeg datuma (**Date** je funkcija koja vraća sistemski datum zapisan u računaru). Ako izvršite naredbu **? d** na ekranu će se ispisati nešto poput **24.10.91**. Sa datumima se mogu izvoditi i neke računske operacije, slično kao sa brojevima. Na primer

```
d2 - d1-> Broj dana izmedju dva datuma.
d + N  -> Datum N dana posle datuma D.
d - N  -> Datum N dana pre datuma D
```

Sve začkoljice koje bi pri tom morali sami da rešavate *Clipper* automatski radi za vas: kontrola meseca, prestupne i proste godine i sl.

U programu se ne mogu direktno pisati konstante tipa datum, što je priličan nedostatak. Na primer ne može se pisati nešto poput **d := 21.01.91**, već se datum mora dobijati kao rezultat izračunavanja neke funkcije ili izraza. Jednu od takvih funkcija smo već upotrebili: **Date()** vraća sistemski datum. Na sreću postoje funkcije za konverziju između stringova i datuma čijim korišćenjem poništavamo ovaj nedostatak. Tako funkcija **CtoD** "pretvara" string u datum, a **DtoC** datum u string. Tako ćemo, ako želimo da promenljivu postavimo na neki određeni datum, napisati

```
d := CtoD ("26.01.66")
```

Slično kao i za brojeve, i ne datume možemo primenjivati ++ i -- operatore. Na primer

```
NewDate := ++CurDate + NoDays
```

Mogu se primenjivati i operatori tipa += -=

```
MyDate +  30
```

Značajne su funkcije koje omogućavaju izdvajanje pojedinih elemenata datuma

```
Day (d)   -> Dan u mesecu datuma D
Month (d)-> Broj meseca u godini datuma D
Year (d)  -> broj godine u datumu D
```

Interesantna je i funkcija za određivanje dana u nedelji u koji pada određeni datum: **DoW**. Treba zapamtiti da ove funkcije "broji" dane na sledeći način: nedelja, ponedeljak, utorak, sreda, četvrtak, petak i subota. Na primer **Dow (CtoD ("11.12.91"))** vratiće broj 4, ovaj dan je bio u sredu.

Datumi se ne moraju pisati u formatu **dd.mm.gg** – način pisanja se podešava naredbom **SET DATE**. Moguć je izbor između nekoliko najpoznatijih načina označavanja datuma, a nas su najinteresantniji engleski i nemački način: **dd/mm/gg** i **dd.mm.gg**. Znači na početku programa obavezno treba staviti **SET DATE TO BRITISH**. Ukoliko vam ne odgovara ni jedan od ponuđenih načina obeležavanja, možete zadati svoj opcijom **FORMAT**

```
SET DATE FORMAT TO "dd,mm,yyyy"
```

Pri tome **DD** označava dan, **MM** mesec, a **YYYY** godinu u datumu. Posle ovakve naredbe može se pisati

```
d := CtoD ("11,12,1991")
```

Sve naredbe će raditi u skladu sa ovom promenom formata.

LOGIČKI TIP I OPERATORI POREĐENJA

Promenljiva **logičkog** tipa može imati vrednosti **T** (tačno) ili **F** (netačno). Vrednost logičkog tipa se dobija izračunavanjem vrednosti nekog logičkog izraza. Najjednostavniji takav izraz je jednostavno poređenje primenom binarnih relacionih operatora. Na primer

```
a == 123
"ABC" == x
x <= y
1 >= 23.1
x <> y
x1+1 < x2
y < z
```

Svaki od pobrojanih izraza demonstrira primenu jednog operatora poređenja: == (jednako), < (manje), > (veće), <= (manje ili jednako), >= (veće ili jednako), <> (različito). Vrednost svakog od ovih izraza je **T** ili **F**. Mogu se porediti brojevi, stringovi i (što se često zaboravlja) datumi. Tako se može pisati

```
Date() + 5 < CtoD("31.12.91")
```

Mogu se porediti i logičke vrednosti, mada se to dosta retko koristi. Pri tome je

```
.T. == .T.
.F. == .F.
.F. < .T.
```

Na logičke izraze se dalje mogu primenjivati logički operatori ! (negacija), **AND** (operator "i") i **OR** (operator "ili") sa uobičajenim značenjem. Na primer

```
(0 <= X  AND, X <= Y)  OR  ! Eof(
```

Zagreda imaju uobičajenu ulogu. Ako niste sa svim sigurni u prioritet operacija najbolje rešenje je korišćenje zagrada.

Logički izrazi se upotrebljavaju na uobičajenim mestima u kontrolnim strukturama programa za proveru uslova. Na primer

```
IF X <> 0
   WHILE X < 100
      ? X++
   END DO
END IF
```

Promenljive su u *Clipper*-u nešto specifičnije nego u standardnim programskim jezicima. Promenljive se ne deklarišu i svake promenljive može imati vrednost proizvoljnog tipa. Tako je na primer savršeno legalno

```
X  = 100 45
? X + 300
X  = "ASCO"
? Left (X  2)
```

To znači da promenljive može sadržati vrednost proizvoljnog tipa – u jednom trenutku to može biti string a u sledećem datum. Tip promenljive se određuje na osnovu njene vrednosti. Ipak ovu osobinu ne bi trebalo zloupotrebljavati, već je preporučljivo na osnovu mena promenljive znati njen tip u bilo kom delu programa, bez razgledanja poseika programa. Di rekina losa posledica je da se za vreme prevođenja ne može kontrolisati valjanost izraza e + 3 je ispravan i zraz ako A ima numeričku vrednost. al je zato izraz "ABC" + 3 tatalna greška. Jedini tek protiv ovoga je dosledno programiranje. uostalom eko se ne možete samostalno disciplinovati neće vas na to nateratl ni me kakve stroge deklerecije! Ipak ova osobina *Clipper*-a u praksi retko stvara probleme. kod lole iskusnijeg korisn ka.

Znači promenljive po tekućoj vrednosti del mo ne različite tipove brojeve stringove. datume. Svaka *Clipper* promenljiva os m toga ima jednu nepromenljivu karakterističku koja se naziva **memorijske klasa**. *Clipper* poznaje **statičke** i **dinamičke** promenljive. Statička promenljive se olormljavaju (definiše se memorijski prostor) za vreme prevođenje programa. Dinam čke se stvaraju po potrebi za vreme izvršava nja programa. Pošto se dinamičke promenljive moraju kreirati pre upotrebe a i pristup njima je dosta kompli kovanji od statičkih promenljivih. rad sa njima je osat no sporiji zato kad god je to moguće treba koristiti statičke promenljive.

Statičke promenljive mogu biti klase **LOCAL** i **STATIC**. **LOCAL** promenljive se definišu za vreme prevođenja programa i kreiraju se po ulasku u procaduru u kojoj su definisane i vidljive su samo u toj procaduri – ni jedna procadura ili funkcija koja se poziva iz te procadure ne može pristupiti direktno lokalnoj promenljivoj. Promenljive tipa **LOCAL** se more deklarisati na početku modula naredbom **LOCAL** <lista imena>. Ova promenljive treba najviše koristiti. one zadovoljavaju uslov potpunog "skrivanja informacija" koji bi trebalo de zadovoljava svaka procedura u programu.

```
PROC DemoLocal
   LOCAL x  = 1, d  = Date()
   ? x++, d
   ProcP2 ()
RETURN
```

U ovom primeru se deklarišu dve lokalne promenljive **X** i **D** i istovremeno se inicijalizuju. Procedura **ProcP2** ne može pristupiti ni jednoj od ove dve promenljive. Inicijalizacija ovih promenljivih se obavlja u-vek po ulasku u procedaru **DemoLocal** to znači da će se **X** i **D** uvek znova postavljati na vrednost navedene u inicijalizaciji kada se aktivira procedura **DemoLocal**. Primetite da se u inicijalizaciji mogu koristiti pro zvolji ni izrazi. ne samo konstante. Tako će prethodni primer uvek spisati broj 1 i tekući datum.

Promenljive klase **STATIC** se takode definišu pri prevođenju. ali se za razliku od **LOCAL** promenljivih ne inicijalizuju uvek kad počne izvršavanje funkcije u kojoj su definisane. Deklarišu se naredbom **STATIC** <liste imena>.

```
PROC DemoStatic
   STATIC x  = 1
   ? x++
RETURN
```

Promenljive **X** sada postoji i čuva prethodnu vrednost i kada procedura **DemoStatic** nije aktivirana. Inicijalizacija se vrši samo jednom i to pri prevođenju programa – zato se kao izrazi za inicijalizaciju moraju upotrebiti konstantni izrazi. Prethodni primer će pri svakom pozivu ispisati **X** uvećan za 1 1 2

I ovakva promenljive su vidljive samo u modulu u kome su definisane. slično lokalnim procedure koje se pozivaju "ne v da" promenljive tipa **STATIC**.

Nešto se razlikuje slučaj ako se promenljiva deklariše kao **STATIC** na početku datoteke izvan svih procedura i tada je ona vidljiva u svim procadurama u toj datoteci – jako pogodno za "interne" globalne promenljive na nivou jednog modula. Naredbe **STATIC** i **LOCAL** se moraju pisati pre svih izvršnih naredbi u modulu u kome se nalaze.

Dinam čke promenljive se kreiraju **izvršavanjem** naredbi **PUBLIC**. **PRIVATE** ili naredbom dodeljivanja za vreme izvršavanja programa.

Globalne promenljive se kreiraju za vreme izvršavanja programa naredbom **PUBLIC** i tekve promenlji va su vidljive iz svih procedura u programu. Promenljive tipa **PRIVATE** se takode kreiraju za vreme izvršenja programa. a v dljive su u procedun u kojoj su definisane i u svim procaduram koje se pozivaju iz te procedure. Ako se u nekoj procaduri navede naredba dodeljivanja **X** := <izraz> a pri tom **X** nije prethodno delin sana. automatsk će se otormit promenljive **X** tipa **PRIVATE**.

```
PROC DemoDyn
   PRIVATE prX     1
   PUBLIC puY     2
   FF ()
   ? prX, puY
RETURN
```

```
PROC FF
   PRIVATE prZ := 3
   prX++ ; puY++
   prW  = 4
RETURN
```

U prethodnom primeru u procedun **DemoDyn** promenljiva **prX** je vidljive u procedurama **DemoDyn** i **PP** a promenljiva **puY** iz svih procedura u programu. Promenljive **prZ** i **prW** (automatsk se kre ira naredbom dodeljivanje) su vidljive samo u procedun **PP** i u procedurama koje ona poziva.

Osim što je rad sa ovakvim promenljivim bitno sporiji od rada sa statičkim promenljivim vrlo je tesko odrzati nezavisnost pojedinih procedura od ostatka programe. Naime. šta vredi eko je procedure napisana bez greška kad sve procedure koja ona poziva mogu nesmetano de menjaju sve promenljive klase **PRIVATE** u ovoj procedun – to znač da pregledom procedura ne možete reći da li će ispravno raditi, to možete tvrditi tek kad pregledate i sve podređene procedure!

NIZOVI

Nizovi u *Clipper*-u se prilično razlikuju od standar dnih nizova koje ste sreli u drugim jez cima. Niz u *Clipper*-u je uređen skup od najviše 4096 elemenata. Nizovi mogu biti ista memorijske klasa kao i promenlji ve **STATIC**. **LOCAL**. **PRIVATE** i **PUBLIC**. Bez obzira na klasu. tip pojedinih elemenata moze biti proizvoljan. Štaviše. čak ni u okviru istog niza svi elementi ne moraju b ti istog tipa. Na primer. prvi element niza može biti datum, drugi string itd. Nizovi mogu imati proizvoljan broj dimenzija a maksimalan broj elementa po jednoj dimenziji je 4096. Niz se deklariše kao i obične promenljive jednom od naredbi za deklarisanja. Na primer. naredba **LOCAL X[10]** definiše lokalni niz **X** od deset elemenata. Pojedinačnim elementima nizova se pristupa na uobičajen način. Na primer **x[2]** := **x[4]** * **x[3]**. N-zov sa više dimenzija se deklariše na analogan način. navođenjem maksimalnih dimenzija svih indeksa. Na primer **PUBLIC X[10,20]** deklariše matricu od 200 elemenata. Element niza ne more biti "obične" vrednosti – element niza može opet biti niz. Tako se može pisati **x[2]** := **34** ali i **x[2]** := **{1, 2, 3}** pri čemu se u drugom primeru elementu **X[2]** dodeljuje niz od 3 elementa. N zovima se vrednosti mogu dodeljivati i direktno. slično kao u prethodnom primeru

```
[1]; 21, {31, 32, 33}
```

Ovde je **X** niz sa tri elementa, pri čemu je treći e lement niz **{31 ,32, 33}**. Na primer, vrednosti 32 se pri stupa kao **X[3][2]** ili **X[3,2]**

Bitno je uočiti da je niz u *Clipper*-u u mnogo "dinamičniji" nego u drugim jezicima. teško je naći pro gramski jezik koji bi "progutao" ovako nešto

```
                    , Date()
```

Ipak. baš zbog ovakvih mogućnosti treba biti o prezan sve operacije sa nizovima se odvijaju dinamič ki i potencijalno su doste spore. N-zovi se u praksi veoma retko upotrebljavaju. ma kako to čudno izgle dalo. Obično se upotrebljavaju za zadavanje nizove opcija i slično. ei veoma retko za neke druge svrhe.

PROCEDURE I FUNKCIJE

Clipper poznaje sasvim standardne procedura i funkcije. Vrlo često ćete pročitati de *Clipper* i ne pravi veliku razliku izmedu ove dva termina. ali to nikako ne treba zloupotrebljavati. već treba koristiti procedure i funkcije baš za one svrhe za koje su i namenjene. procedura imaju proizvoljan broj parametara i nemaju povratnu vrednost, dok funkcije imaju proizvoljan broj parametara i vraćaju tačno jednu vrednost.

Procedure se započinju sa **PROCEDURE** <naziv> [(<lista argumenata>)] a završavaju se sa **RETURN**. Umesto **PROCEDURE** može se pisati samo **PROC**. Na primer

```
PROC Proba (String, Ch1, Ch2)
   LOCAL Tmp
   Tmp := StrTran (String, Ch1,
   IF !Empty (Tmp)
      ? Tmp
   END IF
RETURN
```

Ovako napisana procedura se poziva sa **Naziv** [(<lista argumenata>)]. Na primer **Proba ("ABCEDA", "E", "-")**. Ukoliko procedura nema argumenata, moraju se pisati zagrade **Proba ()**. Funkcije se pišu sasvim slično, jedina razlika je što funkcije vraćaju vrednost nakog tipa. Na primer

```
FUNC Nth (X, N)
   LOCAL Ret
   IF Len(X) > 1 .AND. N
      Ret := X[N]
   ELSE
      Ret :=  F.
   END IF
RETURN Ret
```

Naredba **RETURN** je izvršna naredba i može se naći bilo gde u telu funkcije. Njenim izvršavanjem se prekide tekuća funkcija i vraća se vrednost u glavni program. To znači da naredbu **RETURN** ne morate pisati na kraju funkcije – možete p sati više takv h naredbi unutar tela. *Clipper* odreduje fizički kraj funkcije na osnovu sledeće deklaracije funkcije ili tako što je došlo do kraje datoteke. Ipak. zaista je ružno videti "sakatu" funkciju. mozda je najbolje postupati kao u prethodnom primeru. umesto pozivenja naredbe **RETURN** rezultat se čuve u privremenoj promenljivoj a tek na kraju se izvršava naredba **RETURN**. Sve ovo važi i za procedure i za funkcije. Primetite da funkcija ne mora vraćati vrednost jednog određenog tipa – jedna funkcija u zavisnosti od ulazn h veličina može vraćati vrednosti proizvoljnih različitih tipova. baš kao i prethodnom primeru. Interesantno je da funkcije može vratiti vrednost bilo kog tipa – pa znači i niz. kodni blok. objekt.

Funkcije obično pozivate unutar nekog izraza na uobičajen način. Na primer. prethodnu funkciju možemo upotrebiti u naredbi

```
X := {1, 2,
? X := Nth (X,
```

Značajno je da se funkcije mogu upotrebljavati i kao procedure na određeni način. ako nam nije važno kakva je povratne vrednost funkcije. funkcija se može pozivati i kao procedure. Primer je standardna **SelCo-**

ker funkcija koja kao argument prima opis boje koji treba postaviti a vraća prethodno aktivnu odju. Na primer postavljanje nova boje i čuvanje prethodne obično se piše ovako

```
SavColor := SetColor )"w/r"(
```

Ako nam nije važno (je često nije) kakva je definicija prethodno važeće napisaćemo samo

```
SetColor ("w/b")
```

Pri tome vrednost koju funkcija vraca nema nikakvog značaja – u ovom slučaju je važan samo njen sporedni efekat.

PROSLEDIVANJE PARAMETARA

Važna stvar kod procedura i funkcija je prosledivanje parametara. Prosleđivanje parametara u *Clipper*u je veoma specifično i može da pravi velike probleme ukoliko vam nije do kraja jasno. Prvo, vrlo je važno da znate koji je tip promenljive koju prihvata procedura kao parametar. Na primer u sledećoj proceduri

```
PROC Proba (X, Y)
   ? X, Y
   PotProba()
   ? X, Y
RETURN
```

promenljive **X** i **Y** su **LOCAL** promenljive i procedura **PotProba** im ne može pristupiti i obe naredbe za štampanje odšlampace istu vrednost bez obzira na naredbe koje se izvršavaju unutar funkcije **PetProba**. *Clipper* je od ranijih verzije zadržao još jedan način za prihvatanje parametara pomocu **PARAMETERS** naredbe. U tom slučaju se kreiraju promenljive **PRIVATE** tipa. Tako ako se prethodni primer napiše ovako

```
PROC Proba
   PARAMETERS X, Y
   ? X, Y
   PotProba()
   ? X, Y
RETURN
```

promenljive **X** i **Y** biće tipa **PRIVATE** pa će procedura **PotProba** moći da menja njihove vrednosti. Ovo može rezultovati greškama koje se strahovito lako otkrivaju i na jo trabo strogo obratiti pažnju. bolje je prihvacene vrednost alkoprati u promenljive za koje znamo gde su vidljive nego nagadati ko šta menja.

Sledeća stvar se bča broja formalnih (pn definisanju procedura) i stvarnih parametara (pn pozivu procedura). Ovaj broj sa **ne mora** slagati. Na programeru je da ovu osobinu previlno upotrebi. Pošto tipovi parametare mogu biti potpuno različiti vrlo lako se može desiti da se ono što se šalje i što procedura očakuje ne poklapa. e pravodilac ni ne koji način ne vrši (u slučaju *Clipper* a i ne može vršiti) kontrolu parametara. Zato je preporučljivo uvek vršiti kontrolu broja i tipova prosledenih parametara u procedurama koje imaju vi še argumenata.

Funkcija **PCount()** vraća broj argumenata koji su prosledeni lakućoj procedur. Na primer

```
PROC Message (Code, Text, Comment
   bo CASE
      CASE PCount() == 3
         ? "]" + Code + "}", Text
", Comment
      CASE PCount() == 2
         ? "]" + Code + "}"
      CASE PCount() == 1
         ? "Error code:"
      OTHERWISE
        ? "No Errors"
   END CASE
RETURN
```

Sade se procedura **Message** može ravnopravno pozivati se

```
Message (ErrCode, "Undefined varia
ble", CurVar)
Message (ErrCode, "Undefined
variable")
Message (ErrCode)
Message ()
```

Ovo se vrlo zgodno može iskoristih ze definisanje procedura u kojima se manja broj parametara. Skoro sve komplikovanije funkcije *Clipper* a koriste ovu mogućnosi i dozvoljavaju opcione parametre ako ih ne navedete koriste se neke podrazumevane vrednosti.

Sledeća važna stvar je način prenosa parametara u procedure. Parametri se prenose po vrednosti pre vi se kopija originalnog parametra i evenjualno menja nje parametara u procedun ne menja promenljive u glavnom programu. Na primer

```
PROC Main
   X := "ABC"
   Subroutine (X)
   ? X
RETURN

PROC Subroutine (String)
   String := "EFG"
RETURN
```

Ovaj program štampa nepromenjenu vrednost promenljive **X** 'ABC" jer se pre izvršenja naredbe **String** = "**EFG**" kopira stara vrednost u privremenu promenljivu **String**.

Često je potrebno da procedura ipak menja svoje parametre najčešce kod velikih stringove ili memo polja. Tada se parameln mogu poslati po adresi u tom slučaju se ne prenosi vrednost već **adrese** parametra pa se ave izmene obavljaju na originalnim promenljivama i u glavnom programu. Prenos parametara po adresi se označava simbolom **@** (majmin) pre imena promenljiva. Na primer ako prethodni primer napišemo ovako

```
PROC Main
   X := "ABC"
   Subroutine (@X)
   ? X
RETURN

PROC Subroutine (@String)
   String := "EFG"
RETURN
```


u proceduru **Subroutine** biće prenela adresa stringe **X** pa ča naredba dodeljivanje dodelih novu vrednost. Teko će biti odštampane vrednost "EFG"

Izuzetak od ovog pravila su nizovi. Naime ako pri normalnom prenosu parametara (po vrednosti) pokušata da promenite ceo niz naredbom poput **Array** = **2** i tako niz pretvorite u običnu promenljivu nećete uspeti – ako se prenos odvija po vrednosti naziv niza koji predstavlja adresu niza u memoriji kopira se u privremenu vrednost i vi menjate samo privremenu kopiju, e ne pravi niz. Nizovi su u stvari. realizovani kao pokazivači – naziv niza je promenljive koja čuva adresu početka niza u memonji. Izmena kopije adrese ne proizvodi nikakav efekat. Ali ako se u proceduri nade naredba popu **Array[2]** = **3**, bez obzira što je napravljena kopija pokazivača na originalni niz, ona pokazuje na **isti** objekat tako da će prethodna naredba izmeniti originalni niz bez obzira na način prenosa.

PROŠIRENI SKUP NAREDBI

KONTROLA EKRANA

Značajan deo u svakom *Clipper* programu čine procedure za komunikaciju sa korisnikom. Zalo je čitav deo osnovnog *Clipper* jezika posvećen upravo naredbama za kontrolu ekrana.

Dosta naredbi se odnosi na ispisivanje vrednosti na ekran. Jednu naredbu za ispis vanje smo već koristili "?" Ove naredba ispisuje vrednosti proizvoljnih zadatih promenljivih a pre toga pomera kursor u sledeći red. Znači ispisivanja uvek počinje od početka sledećeg reda u odnosu na tekuću poziciju kursora. Ova naredba je sinonim za funkciju **QOut** koja ima potpuno isti efekat. Na primer

```
? x, y + 2, "ABCD", Date()
QOut (x, y + 2, "ABCD", Date())
```

Slične je i naredba "??", koja se razlikuje samo po tome što pre ispisivenje na pomera kursor – ispisivanje počinje od tekuće pozicije kursora.

Pozicija kursora se menja ili ispisivanjem ili funkcijom **SetPos (x, y)** gde su **X** i **Y** nove koordinate kursora. Trenutna pozicija kursora može se pročitati funkcijama **Row()** i **Col()** – ova vraćaju redom red i kolonu u kojoj se nalazi kursor. Numerisanje počinje od gornjeg levog ugla ekrana (koordinata (0 0)) i kreće se do (24 79) – poslednji simbol poslednje linije na ekranu.

Ispisivanje sa obično obavlje bržom naredbom **SAY**

```
ex" # " 5.01"
```

Ova naredba ispisuje string **"Clipper5 01"** na koordinati (3 10) na ekranu.

Sastavni deo estetskog ubličavanje programa je i crtanje raznih pravougaonika sastavljenih od grafičkih ambola iz ASCII tabele. Bez nepomog nagađanja kodova i beskonačnih kombinacija **<Alt>xxx>**. pravougaonik se jnvijalno crta sledećim naredbama

```
UBLE
```

Prva naredba crta pravougaon k sa specificiranjem koordinatame jednostrukom a druga dvostrukom linijom. Za "mešanje" jednostrukih i dvostrukih okvira pogledajte opis naredba BOX.

Brisanje celog ekrena izvodi se naredbom **CLS**. Brisanje prevougaonike na ekranu obavlje naredba

```
12,50
```

UČITAVANJE VREDNOSTI I MENIJA

Vrlo važan deo *Clipper*-a se odnosi na unos vrednosb koje korisnik zadaje preko tastatura.

Najjednostavniji oblik predstavlje naredba **MENU TO**. Ona omogućava izbor izmedu nekoliko ponuđenih opcija.

```
SET MESSAGE TO 24 CENTER
@ 0,0 PROMPT "Upis" MESSAGE "Upis po
dataka"
@ 1,0 PROMPT "Izmena" MESSAGE "Izmena
podataka"
@ 2,0 PROMPT "Brisanje" MESSAGE "Bri
sanje podataka"
MENU TO Choice
DO CASE
   // Pritiseut je ESC
   CASE Choice == 0 , RETURN
   CASE Choice == 1 ; Add ()
   CASE Choice == 2 , Chg ()
   CASE Choice == 3 ; Del ()
END CASE
```

Prvo se naredbama **PROMPT** specificiraju opcija menija. koord nate gde ih treba ispisati i opciona dodatna objašnjenja. Zatim se naredodm **MENU TO** aktivira meni i korisnik može da bira jednu od ponuđenih opcija ili da pritisne jaster **<Esc>** čime pon štava izbor. Neredba **SET MESSAGE** služi za podešavanje linije u kojoj će se ispisivati dodelna objašnjenja o stavkama menija. Slično dejstvo, ali uz mnogo veće mogućnosti možete postići funkcijom **AChoice**. Morate znati da ove funkcije sluze za kreiranje korisničkog interfejsa samo na najnižem nivou. Pra nekoliko brojeva u "Računanma" je objavljen kompletan korisnički interfejs koji vam može poslužiti kao osnova za veše programe.

Kada vam u nakom programskom jeziku zatreba učitavanje vrednosti obično je izvodile krajnje primitivnim funkcijama jer je potpunja kontrola ulaza skopčana sa pisanjem dosta složenih procedura. *Clipper* ma u osnovnom jeziku veome fleksibilen i moćan sistem koji vas skoro potpuno rešava ovih problema. Takozvani **GET** sistem učitave proizvoljne vrednosti uz potpunu kontrolu i mogućnosi proizvoljen zmene. Ako je na primer potrebno učitati podatke za jednu osobu to se može uraditi ovako

```
Ime := Space (20)
Adresa := Space(30)
DatRodj := Date()
BrClana := 0
```

```
@ 1,0 SAY "IME"      GET Ime
@ 2,0 SAY "ADRESA:"  GET Adr
@ 2,0 [illegible]    GET [illegible]
@ 3 [illegible]
READ
```

U ovom primeru se prvo naredbama **SAY..GET** zadaju promenljive koje treba učitati i kratki opisi – objašnjenje ispred svake vrednosti. Zatim naredba **READ** aktivira specifičnu masku za unos i korisnik može proizvoljnim redosledom da unosi tražene vrednosti. Kada pritisne **<Esc>** ili **<Enter>** na poslednjoj vrednosti, unos se završava i promenljive dobijaju vrednosti koje je korisnik uneo. Jednostavno i efikasno. Ovo je samo najjednostavniji nivo korišćenja **GET** sistema. Svaka od **SAY..GET** naredbi ima niz opcija za potpuniju kontrolu učitavanja.

Najjednostavnije je promeniti boju opcijom **COLOR**. Ona vam omogućava da promenite boju kojom će biti ispisana bilo koja od vrednosti (ispisivanje se inače izvodi podrazumevanom bojom koja se zadaje funkcijom **SetColor**).

Opcija **VALID** omogućava specifikaciju uslova koji mora biti zadovoljen da bi uneta vrednost bila korektna.

```
POZITIVAN BROJ " GET x VALID x > 0
```

Tako korisnik mora uneti korektnu vrednost. Na ravno uslov može biti proizvoljno komplikovan.

Opcija **WHEN** omogućava kontrolu nekog uslova, ali pre nego što korisnik počne sa unosom vrednosti promenljive – na taj način se može preskočiti unos nepotrebne vrednosti ili realizovati unos z nekog mesta vrednosti. Sledeći primer po izlasku u polje automatski otvara meni:

```
PROC [illegible]
  @ [illegible]
Vals[1]
  "
  READ

RETURN

FUNC Val
   @ 1,0 PROMPT "DINAR"
   @ 2,0 PROMPT "MARKA"
   MENU TO Val
RETURN .T.
```

Ovim mogućnosti **GET** sistema nikako nisu iscrpljene – postoji mogućnost potpune kontrole uz veoma specifična dotjerivanja. Za više detalja pogledajte originalnu literaturu – **GET** sistem je posebna priča koj ovde ne možemo posvetiti veću pažnju.

BOJE

Važan detalj pri izradi programa predstavljaju boje. *Clipper* omogućava odličnu kontrolu boja i svakako je treba do kraja iskoristiti kako bi vaši programi bili efektniji. *Clipper* poznaje nekoliko raz ličjih boja u zavisnosti od tipa grafičke kartice.

BOJA	OZNAKA
CRNA	N
BELA	W
PLAVA	B
ZELENA	G
CRVENA	R
CIJAN	BG
LJUBIČASTA	RB
BRAON	GR

Naravno, na monohromatskom monitoru mogu se koristiti samo bela i crna boja i kao dodatak može se uključiti atribut podvlačenja oznakom **U**. Za svaku od ovih boja postoji i svetlija varijanta koja se označava plusom **+** označava svetlo plavu odnosno **GR+** svetlo braon (žuta). Još može se koristiti i oznaka ***** koja uključuje treptanje. **G*** označava zelenu boju koja treperi.

Da bi se boja promenila, mora se zadati nova boja slova i boja pozadine. To se radi konstrukcijom **f/b** gde je **F** boja slova, a **B** boja pozadine.

Kada se promeni boja, sve ispisivanje na ekran će se nadalje obavljati izabranom bojom. Ža promenu aktivne odje koristi se funkcija **SetColor** koja postavlja novu boju slova i pozadine. Na primer **SetColor ("W/F")** menja boju u belu slova ne crvenoj pozadini. Stvar ipak nije tako jednostavna. Pošto svi podsistemi *Clipper*a pošljuju podajke o tekućoj boji, ne bi bilo zgodno da se sve ispisuje istom kombinacijom odja. Zato se pri zadavanju nove boje mora zadeti 5 kombinacija razdvojenih zarezima. Ne primer

```
SetColor ("W/n, x/g, n/w, , w/b"
```

Prva boja je najvažnije i koristi se za sve funkcije koje ispisuju vrednosti na ekran **SAY QOut @ TO**. Ze prethodni primer to je **W/N** bela slova na crnoj pozadini.

Druga boje se koristi za izabrane elemente menije i izabrane vrednosti u **GET** sistemu – ovu boju obavezno postavite da bude različita od prva kako bi korisnik jasno video šta unosi.

Treća boja označava boju okvira – ova boja ime značaja samo na VGA karticama.

Četvrta boja se ne koristi i možete je preskočiti pri specifikaciji.

Peta boje se odnosi na neodabrane elemente menije i vrednosti u **GET** sistemu. One se obično postavlja de bude ista kao prva boje.

Početnici obično prave ogromnu grešku i funkciju **SetColor** koriste po potrebi po celom programu. To rezultuje sjrahovitim ješkocama pri pokušaju de se promena boje u programu "juriti" boje po programu nije nimalo zabaven posao. Osnovno pravilo je ba se sve boje koja se koriste definišu na početku programa kako bi se kasnije izmene vršile na jednom mestu. Dobro je predvidet da će program raditi i na različitim ji povime monitore. Tip monijora (kolor ili monohromatski) može se ispitati funkcijom **IsColor()** koja vreća **T** ako je na računaru kolor grafička kartica. Obično se u zavisnosti od tipa monitore korišćenjem funkcije **IsColor** poslavljaju odgovarajuće vrednosti boja. Ne primer


```
[illegible]
```

Obično se koristi prva boja, tako da ponekad nije važno postaviti sve odje.

Skoro sve naredbe za ispis na ekran poseduju mogucnost promremene promene boje odja se menja samo za jekuc ispis. Na primer naredba **@ 0,0 SAY Clipper COLOR W/b** spisace poruku belim slovima na plavoj pozadini, bez obzira na takucu boju.

KONTROLA TASTATURE

Clipper ima veoma specifičan i pomalo neobičan sistem za kontrolu tastature. Ovo u prvi mah može iz gledaji neobično, ali ako se zna da je ogromen deo *Clipper* programa obično posvecen komunikaciji sa korsnikom, stvan dolaze na svoje mesto.

Svaki taster koji korsnik otkuca poseban podsistem upsuje u bafer za tastaturu. Frec zmje redom se upisuju **kodovi** tastera koje korsn k pritisne. Za učitavanje prit snutog tastera obično se koris t funkcije **Inkey**. Ona omogućava da se učita samo jedan simbol se tastature baz kakvog isp sivanja na ekranu. Ako se navede naredba **Ch = Inkey(0)** funko-ja čeka dok se ne pritisne nek tasler i tek onda nasjavlja program. e funkcija vraca kod pritisnutog tastera. Čekanje se mo že i skratit. **Inkey(N)** čeka najviše **N** sekundi a posle toga vraća kod pr isnulog tastera. 0 ako n-ista nije pritisnuto. Sledeća varijanta je **Inkey()** – sada funkcija samo prover de li je u međuvremenu stigao nek taster u bafer. Jesture i vraća kontrolu glavnom programu.

Da bi se dobio poslednji tasler koji je korsnik otkucao, koristi se funkcija **LastKey**. One se standardno korsli pn odredivanju de li će se podaci koje je korisnik otkucao upisati u datoteku. Naime, korisn k ima dva načina da izađe iz **GET** maske za unos: prvi je pomoću tastera **<Esc>** i tada se poništavaju sve unjta l zmene pa ne treba ništa upisivati u datoljeku i drugi (na primer končenjem jastera **<Enter>**) kada treba upisati podatke u datoljeku. Tako se u programu često sreće ovakva konstrukcija:

```
IzmeniPodatake ()
IF LastKey () = 27
   UpisPodataka ()
END IF
```

Pr Jome je 27 kod tastera **<Esc>**. Funkcija **LastKey** vraća kod jastera koji je **obraden**, tj. koji je već proizvao odgovarajuće dejstvo – u prethodnom primeru pritisnuj jaster je već obrad o **GET** s stem tako da je ova funkcije u izvesnom smislu "pogled u prošlost". Možamo zaviriti i u budućnost – funkcija **NextKey** vraća kod jastera koji je sledeći na redu za obradu. Ova je kor sna u slučaju neke obrade koja duže traje – jade nekad treba predvideti šta je korisnik u međuvremenu pritiskao. Funkcija **NextKey** ne menja bafer tastature – pročitani jester će se normalno obraditi kad dođe na red.

Kao što je već rečeno veliki deo *Clipper* programa se odnosi na rad sa tastaturom. Obično sa prvo skanira tastatura funkcijom **Inkey**, a zatim se u zavisnosti od pritisnutog tastera preduzima dalje akcija. Mnogo iskusnji korsnici znaju da je pravilna struktura programe nepravilne s]ruktura za korsnika vrlo često se ukazuje potreba za nekim ekvivalentom **GO TO** naredba koje nema u *Clipper* u. Rešavanje ovog problema uvođenjem raznih indikatora je veoma ružno i nepreporučljivo rešenje. Kada bi samo korsnik znao šta je problem i u pravom trenutku otkucao jedan taster! U takvim slučajev me spas nam donosi naredba **KEYBOARD**, koja upisuje zadate kodove u bafer tastature. Znači, glejkaj je ish kao da je korisnik otkucao ono što mi zadamo. Na primer

```
KEYBOARD "[illegible]ABC[illegible]
```

Ova neobično upotrebljiva naredba se najčešće koristi za kontrolisanje raznih izuzetaka u korsničkom interfejsu. Traba imati u vidu da brže prelhodni sadržaj bafera tastature – sve što je korisnik otkucao u međuvremenu nepovratno se briše.

KODNI BLOKOVI

Kodni blokovi (**Code Blocks**) su velika novina u *Clipper* u 5.01. Šta je to u stvari? To su ukratko rečeno specijalne funkcije sa kojima se mogu izvoditi određene operacije kao sa podacima. Kodni blok se može dodeljivati promenljivoj, baš kao i ostali tipovi podataka. Na primer

```
cBlk = {|x,y| x+y }
```

U ovom primeru je definisan kodni blok sa dva argumenta koji vraca njihov zbir – baš kao normalna funkcija. Takav kodni blok se dodeljuje kao vrednost promenljivoj **cBlk**. Ako se malo kasnije napiše **NewcBlk = cBlk**, i promenljive **NewcBlk** će imat kao vrednost isti kodni blok.

Kaba zatreda da se izvrši kodni blok, koristi se funkcija **Eval** – ona prihvata kao argumenje kodni blok i listu njegovih argumenata, a vraća izračunatu vrednost. Na primer

```
? Eval (cBlk, 2, 5)
```

odštampaće vrednosj 5. Kodni blokovi se mnogo češće koriste kad treba neku funkciju primeniti ne sve elemente nekog niza, ili ne sve slogove neke datoteke. U prvom slučaju se koristi funkcija **AEval**, a u drugom **DBEval**. Funkcija **AEval** je veoma pogodna za zbegavanje spore **FOR** petje. Sledeći primer štampa sve elemente niza:

```
cBlk := {|x| QOut (x) }
AEval (Array, cBlk)
```

Kodni blok **cBlk** se izvršava prvo sa argumentom **Array[1]**, zatim sa **Array[2]**, itd. do kraje niza. Veoma slično se kodni blok izvršava za svaki slog aktivne DBF datoteke. U ovom slučaju nema argumenata, već se jednostavno pomere pokazivač datoteke od prvog do poslednjeg sloga, omogućavajući da kodni blok radom pristupa slogovima. Na primer, sledeći kodni blok štampa svaki slog u DBF datoteci sa polji ma **IME** i **PREZIME** i **ADRESA**.

```
SELECT Adrese
cBlk := { || QOut(ime, prezime, adresa)}
DBEval (cBlk)
```

Kad telo kodnog bloka može se navesti bilo koji ispravan izraz – komande se ne smeju koristiti. Da bi se proširio skup mogućnosti, *Clipper 5.01* ima veliki skup funkcija koje imaju isto dejstvo kao odgovarajuće naredbe. Jedna od takvih funkcija je i **QOut**, koja "zamenjuje" naredbu ?. I bez toga je lako pretvoriti niz naredbi u funkciju: prvo treba napisati standardnu funkciju sa željenim nizom naredbi a zatim u kodnom bloku pozvati tu funkciju – na treba zaboraviti da je i funkcija korektan izraz.

Kodni blok može vraćati kao rezultat vrednost proizvoljnog tipa i to je vrednost izračunatog izraza – preciznije – vrednost **poslednjeg** izračunatog izraza u telu kodnog bloka. Naime, kao telo se može navesti i više izraza odvojenih zarezima.

```
cBlk := { |x, y| QOut(x), QOut(), QOut(y), x + y }
```

Vrednost ovog kodnog bloka je **X + Y**, jer je to poslednji izraz u nizu.

Važno pitanje kod kodnih blokova je koje su promenljive vidljive u njihovim izrazima? Na prvi pogled može izgledati logično da su vidljive promenljive iz procedure u kojoj se kodni blok izvršava. Mada izgleda logično to nije tačno i važi jednostavno pravilo: kodni blok je u stvari **deo** procedura/funkcije u kojoj je definisan i za njega su vidljive sve promenljive kao i za proceduru u kojoj je definisan a ne u kojoj se izvršava. U sledećem primeru je to pokazano:

```
PROC Maie
   LOCAL x := 1, y := 2
   cBlk  := { || QOut(x+y)}
   Subroutine (cBlk)
RETURN

PROC Subroutine (c)
   LOCAL x := 3, y := 4
   Eval (c)
RETURN
```

Ovaj program će odštampati 3, a ne 7, kako bi se moglo pretpostaviti iz ovog primera se vidi da kodni blokovi mogu lepo poslužiti kao svojevrsni kanali za komunikaciju između procedura.

S obzirom da su kodni blokovi potpuna novina postavlja se pitanje gde se mogu primenjivati. Još veću nedoumicu stvara postojanje makro operatora kojim se postiže sličan efekat. Kodni blok je deo koda kojim se može manipulisati – može se dodeljivati promenljivoj, proveravati jednakost, funkcije ga mogu vraćati kao vrednost, a može se i izvršavati (primenjivati na određene argumente). Sličan efekat se postiže kombinacijom funkcija-makroa, ali se jednom bitnom razlikom: makroi se interpretiraju za vreme izvršenja programa, a kodni blok je **prevod** koji se kreira za vreme prevođenja a samo izvršava za vreme izvršavanja programa, pa je jako značajno brži. Pošto se kodni blokovi mogu upotrebljavati u izrazima (i sami su ispravni elementi izraza), dobro je upotrebljavati ih svuda gde je to moguće umesto makro operatora, jer su znatno brži. Takođe je preporučljivo da se koristi u izrazima umesto funkcije zbog preglednosti – ako se funkcija koristi samo na jednom mestu, bolje je da je vidljiv i njen kod – uporedite sledeće dve naredbe:

```
Chkx()
@ 5,0 GET x VALID ( || pos := ASCan
(Vale, x), x < pos )
```

Kasnije ćemo videti da su kodni blokovi nezamenljivi kod objekata – tamo predstavljaju "žive" poruke koje se šalju objektima i koje sadrže na podatke već akcije.

OBJEKTI

O objektima u *Clipper*u je bilo reči u ranijim brojevima "Računara". Ovde ćemo se ukratko osvrnuti na njihove osnovne karakteristike.

Objekt je u stvari novi tip podataka. Promenljiva tipa objekt sadrži referencu (adresu) skupa podataka i procedura u memoriji namenjenih nekoj određenoj funkciji. Tipičan primer je tabelarni pregled podataka. Kada se kreira objekat koji služi za tabelarni pregled podataka, u memoriji se rezerviše prostor za interne promenljive i procedure potrebne za red top objekta, dimenzije na ekranu, boje kolona, nalpisi iznad kolona, procedure za prelazak na sledeći i prethodni slog itd. Sve to zajedno čini jedan objekt. Adresa top objekta se dodeljuje nekoj promenljivoj i tada se kaže da je promenljiva sadrži vrednost tipa **objekt** (u stvari sadrži njegovu adresu). Na primer, sledeća naredba kreira objekat za tabelarni prikaz:

```
TB := TBrowseNew (0,0,24,79)
```

Promenljiva **TB** sadrži adresu upravo kreiranog objekta. Dalji rad se kreiranim objektom se zasniva na komunikaciji između programa i objekta: program može menjati neke interne promenljive i procedure objekta, a objekat izvršava svoju ulogu u skladu se stanjem svojih internih promenljivih i procedura. U suštini izuzetno prirodna komunikacija.

Clipper ne dozvoljava da sami kreirate svoje klase objekata – možete kreirati samo objekte koji pripadaju nekoj od ugrađenih klasa za tabelarni pregled (**TBrowse**), **GET** sistem, **Error** sistem i **TBColumn** kla se za definisanje kolona u tabelarnom pregledu.

Internim promenljivim i procedurama objekta se pristupa operatorom ":" (dvotačka). Na primer **TB freeze** će vrednost interne promenljive **freeze** objekta **TB**. **TB stabilize()** aktivira internu funkciju objekta **TB**. Ovakva konstrukcija se tretira kao i svaka promenljiva odnosno funkcije. Može se pojaviti kao potpuno ravnopravan član izraza ili u naredbi dodeljivanja:

```
IF TB:stabilize()
   TB:freeze := 3
END IF
```

Objekti su veoma značajno unapređenje u *Clipper*u. Prednost se naročito vidi u tabelarnom prikazu koji se sada realizuje **TBrowse** objektima a zamena je za **DBEdit** funkciju. Na listingu 1 je dat obrazac procedure koju možete koristiti za tabelarni pregled podataka iz proizvoljne datoteke.

MAKRO OPERATOR

Ako ste nekada programirali na *BASIC*u sigurno se sećate funkcije **VAL** koja je interpretirala proizvoljan

Listing 1

```
#include "Inkey.ch"

// Obrazac za proceduru za tabelarni pregled

PROC NewDbEdit
   LOCAL TB, Ch, X, Y, Acur, ACurC

   // Otvaranje datoteka, postavljanje relacija,   .
   USE Primer INDEX Primer NEW
   // ...

   // Kreiranje TBrowse objekta
   TB := TBrowseDB (x1, y1, x2, y2)

   // Postavljanje parametara
   // Boja
   TB:ColorSpec := "W/N,N/W,,,W+/B"
   // Separator
   TB:headSep  := " "
   // Datalo

   // Definisanje kolona
   // NAZIV_1 je matrica izmad prve kolone
   // EXP_FIELD1 je izraz koji računa vrednosti

   TB:addColumn (TBColumnNew ("NAZIV 1",{|| Exp_Field1}))
   TB:addColumn (TBColumnNew ("NAZIV 2",{|| Exp_Field2}))
   // ...
   TB:addColumn (TBColumnNew ("NAZIV N",{|| Exp_FieldN}))

   // Glavni ciklus
   WHILE .T.

      // Regeneracija ekrana
      Ch := 0
      WHILE !TB:stabilize() .AND. ((Ch := Inkey()) == 0) ; END DO

      // Ekran je ispisan i ništa nije pritisnuto
      IF Ch == 0 .AND. TB:stable
         // Osvetljavanje tekuce linije
         X := TB:RowPos
         Y := TB:ColPos
         ACur  := {X, 1, X, TB:colCount}
         ACurC := {X, Y, X, Y}
         TB:   colorRect (ACur, {5,1})
         TB:   colorRect (ACurC, {2,1})

         // Cekaj taster
         Ch := Inkey(0)

         // Ugasi kursor
         TB:colorRect (ACur, {1,2})
      END IF

      // Analiza pritisnutog tastera
      DO CASE

         // Exit
         CASE Ch = K_ESC .OR. Ch = K_ENTER
            EXIT

         // Ovde ubacite kontrolu vasih komandi
```

```
      CASE Ch == K_F2
           DoF2 ()

      //

      // Pomeranje kursora
      OTHERWISE
         Apply (TB, Ch)

   END CASE
ENO DO

// Zatvaranje datoteka
CLOSE ALL
RETURN


// Procedura koja kontrolise pomeranje kursora

PROC Apply (oT0, Key)

   DO CASE
      CASE Key == K_UP          , TB:Up ()
      CASE Key == K_DOWN        , TB:Down ()
      CASE Key == K_LEFT        , TB:Left ()
      CASE Key == K_RIGHT       , TB:Right ()
      CASE Key == K_HOME        , TB:Home ()
      CASE Key == K_END         , TB:End ()
      CASE Key == K_PGUP        , TB:PageUp ()
      CASE Key == K_PGDN        , TB:PageDown ())
      CASE Key == K_CTRL_PGUP   , TB:GoBottom()
      CASE Key == K_CTRL_PGUP   , TB:GoTop()
      CASE Key == K_CTRL_LEFT   , TB:PanLeft()
      CASE Key == K_CTRL_RIGHT  , TB:PanRight()
      CASE Key == K_CTRL_HOME   , TB:PanHome()
      CASE Key == K_CTRL_END    , TB:PanEnd()
   END DO
RETURN
```

Obrazac procedura za tabelarni pregled podataka iz proizvoljne datoteke

izraz zadat kao niz simbola. Na primer **VAL ("123")** vraća broj 123. Funkcija **VAL** je ju završavala ulogu se mo kod jednostavnijih interpretera neki su omogućavali da se kao argument funkcije **VAL** upotrebi svaki a ntaksno spravan izraz. Na primer, zraz **VAL("X+Y")** bi na osnovu tekuće vrednosti promenljivih **X** i **Y** izračunao vrednost **X+Y**. Pošto se kao elementi izraze mogu javljati i funkcije, znači da korišćenjem ovakve funkcije možemo dinamički (za vreme izvršavanja programa) ulicati na naredbe koje će biti izvršene. Jako što kreiranje string koji predstavlja zraz. Ovakva mo gućnost je maksimalno konščene u svakom iole kom plikovanijem programu i funkcije **VAL** je važila za jednu od najmoćnijih funkcija *BASIC*-a.

Važno je uočiti da je implementacija ovakve fun kcije trivijalna u jezicima koji se interpretiraju - dovolj no je ulaz interpretera "preusmeriti" na argument funkcije **VAL**. Kod prevodilleca je stvar bitno različita i interpretiranje izraze je povezano sa poslojanjem mo dule koji interpretira izraze. Međutim, ovakav modul ni je baš trivijalan, a nije nešto naročito ni potreban u standardn m programsk m jezicima. Međutim, kod ba za podataka postoji veoma specifična komunikacija konsnik računar. Svi mogući slučajevi koji mogu na stati u jakvoj komun-kaciji veoma ješko se mogu svesti na izbor između nekoliko opcija: problem su obično takv de ni konsnici ni programer nisu sasvim sigurni na kakva pitanja će morati da odgovara sistem zadužen za odgovore u bazi podataka. Jed ni način da se ovo reši je **interpretiranje** onoga što kor snik unese kao uslov pretraživanja. Zato je *Clipper* opremljen izv **mekro operatorom** koji dozvoljava interpretiranje izraza za vreme izvršavanja programe. U originalnoj doku mentaciji se navod zraz *run time* prevođenje, ali se u stvari radi o prevođenju u specijalni medu kod koji se interpretira. Poslojanje ovog operatora je bitno uticalo na izmenu strukture čitavog *Clipper* programa. Jako da se ponekad postavlja pitanje da li je *Clipper* prevodilec ili je samo maskirani interpretor? Ovo pitanje za neki "veliki" DBMS sistem može izgledati smešno jednostavno, neke stvari se moraju interpretirati - poslop go m le knjige napisanih o optimizaciji upita nad bazom podataka koji se izvode u vreme izvršavanje progra ma. Kratko rečeno: baze podataka **moraju** imati neku mogućnost interpretacije izraze. *Clipper* je odlično us kladena celina -- u sebi sadrži pomešane osobine pre vodioca i interpretera koje odlično saraduju.

Mekro operator se obeležava simbolom & (And) i ne najjednostavnijem nivou se koristi za pretvaranje stringa u odgovarajući broj

```
Var := "123"
? &Var + 1
```

Dati program će odštampati vrednost 123+1=124. Kao argument se može specificirati i ceo izraz

```
Exp := "Plata * Koef + Nagrada   Dugo-
vi"
? &Exp
```

Prethodni primeri nisu imali nekog praktičnog značaja, pošto je izraz poznat i ze vreme pisanja programa, dovoljno je pisati sam izraz umesto poziva ma kro operatora. Međutim, mekro operator dolazi na svoje u ovakvim primerima

```
ACCEPT "Unesi izraz" TO Exp
? "Vrednost izraza ", &Exp
```

Kada se kao argument makro operatore koristi ceo izraz, a ne samo vrednost jedne string promenlji va, potrebno je izraz ograditi zagradama. Ne primer

```
SET FILTER TO &(UserCond + SystemCond)
```

Mekro operator je značajno moćniji od obične funkcije koja interpretira izraze. Njime se mogu menjati delovi programa uz neka ograničenja. Na primer, u sledećem primeru se naziv niza koji će se kreirati odre duje za vreme izvršenje programa.

```
ArrayName  = "Vector"
&ArrayName  := {1, 2, 3}
? 100 + &ArrayName,[2]
```

Tačka posle **ArrayName** je upotrebljena da bi se izbegla dvosmislenost: da li mekro treba upotrebiti na promenljivu **ArrayName** ili na element niza **ArrayName[2]**. Tačka se može uvek upotrebljavati i dobro je i zbegavati sve moguće slučajeve kada nije potpuno jasno šta će se izračunati.

Mekro operator se često upotrebljave za specifici renje funkcije koju treba pozvati za vreme izvršenje programe. Primer končenja je funkcije **MemoEdit** za editovanje stringa koja kao jedan od argumenata ime naziv korisnikove funkcije koja će kontrolisati rad **MemoEdit** sistema. Sledeći primer pokazuje sličan slučaj

```
PROC AnyProc (ProcName)
   &ProcName ()
RETURN
```

Ako se procedura **AnyProc** pozova sa **AnyProc (ErrProc")** biće izvršena procedura **ErrProc**.

Treba biti umeren sa upotrebom makro operatora zbog interpretacije je znatno sporiji od kodnih blokove s naročito od fiksnih funkcije. Treba napomenuti da mekro operajor ne može da interpretira naredba već izraze, tako da se ključne reči naredbi ne mogu in tarpretrati. Prevodilac mora u fazi prevođenja da zna o kojoj se naredbi radi, kako bi napravio korektan prevod.

PRETPROCESOR

Pretprocesor je deo prevodioca koji omogućava obradu teksta programa za vreme prevođenja. Na jaj način možete realizovati jedan nivo iznad *Clipper* programa, koji se također može isprogramirati. Naredbe koje se zadaju pretprocesoru zovu se *direktive*.

Pretprocesor ugrađen u *Clipper* ima standardne direktive koje imaju i svi klasični pretprocesor: **#define #ifdef #ifndef #undef** i **#include**.

Direktiva **#include** služi za uključivanje datoteka u tekuću - efekat je isti kao da se specificirana datoteka umesto na mesto **#include** direktive. Ovo je izuzet no korisna mogućnost koje služi za jednostavno uključivanje čestih definicija u program.

Direktivom **#define** mogu se definisati konstante čime se jako poboljšava preglednost programa. Na primer, umesto da svuda pišete **IF LastKey() == 27** napisaćete na početku programa

```
#define ESC 27
```

i onda u celom programu pišete **IF LastKey() == ESC**, što je neuporedivo preglednije. Uz *Clipper* se i sporučuju nekoliko standardnih datoteka se definicija me od kojih je najkorisnija **INKEY.CH**. One sadrži definicije svih tastera koji nam trebaju pri obradama.

Direktiva **#command** je izuzetno moćna, ona o mogućava korisniku da jednostavnim i prirodnim zapi som kreira svoje naredbe koje će biti potpuno ravnopravne sa ostalim naredbama *Clipper*-a! Ne pri mer, sledeće direktive dodaju **REPEAT UNTIL** petlju osnovnom jeziku

```
#command REPEAT => WHILE .T.
#command UNTIL <lExpr> => IF <lExpr>
; EXIT , END ; ENO
```

Posle ove direktive pretprocesor svako pojavljivanje komande **REPEAT** zamenjuje tekstom "**WHILE .T.**" što je specificirano prvom direktivom. Druga direktive specifica de se konstrukcija **REPEAT-UNTIL** zameni nizom naredbi koje proveravaju uslov za kraj ciklusa. Posle ovih definicija u programu se može pi sati

```
REPEAT
   ? X
UNTIL X <  100
```

potpuno ravnopravno sa ostalim komendama! Da ovo nije puka igrarija, pokazuje to da su sve naredbe novog *Clipper*-a i realizovane preko pretprocesorskih direktiva. I najlepša stvar: sve definicije se nalaze u da toteci **STD.CH** i **potpuno su otvorene za izmene!** To znači da ne samo što možeta da definišete svoje nove komande koje su ravnopravne sa ostalim već da možete potpuno legalno da izmenje i osnovni jezik po ličnim potrebama. Ako malo pažljivije pogledate datoteku **STD.CH** videćete da se ceo *Clipper* jezik sastoji od svega nekoliko osnovnih konstrukcija a sve ostalo su u stvari pozivi funkcija maskirani pretprocesorskim direktivama.

Pri pisanju komendi složenije struktura ponekad je korisno pogledati rezultat rade pretprocesore. Što se

postiže opcijom /P pri prevođenju. Tako se generiše PPO datoteka koja sadrži izlaz pretprocesora.

Primenjena koncepcija omogućuje da se nove verzije *Clipper*-a na nivou sintakse jezika realizuju samo kao datoteke pretprocesorskih direktiva, potpuno nezavisno od "pravog" prevoda koji *Clipper* generiše.

BAZE PODATAKA

Kao što vidite, naredbe za kontrolu toka i uopšte osnovna koncepcija *Clipper*-a je vrlo standardna i logična. Zapitaćete se, u čemu je uopšte specifičnost *Clipper* a? Neke od razlika smo videli u drugom delu ovog teksta. *Clipper* se od klasičnih jezika razlikuje po jome što je namenjen pisanju programa za upravljanje bazama podataka. Zadržaćemo se na trenutak na ovom pojmu. Baz obzire što se o bazama podataka često govori programeri obično imaju vrlo naivne predstave o njima. Zato se osnovni termini upotrebljavaju u vrlo sumnjivim pa i potpuno neodgovarajućim situacijama. Naravno, *Clipper* se može koristi bez ikakvog znanja iz ove oblasti- ali pre ili kasnije ćete (prekoo skupih i mukotrpnih eksperimenata) doći do nečega što je davno utvrđano. Zato je dobro razlikovati bar osnovna usvojene termine.

BAZA PODATAKA

Šta je to u stvari **baza podataka**? Ako zaobiđemo stroge definicije, može se reći da je baza podataka veliki skup podataka o nekom realnom objektu. Moram priznati da ovakva definicija nije baš preterano sikovita – ali zaista je teško naći razumljivu bez ulaženja u teorijske specifičnosti. Mi ćemo se koncentrisati na konkretne primere. Na primer, ako automatizujemo poslovanje jedne radne organizacije mi podatke koji su vezani za poslovanje predstavljamo nekim brojevima i simbolima u dajotekama. Naravno, podaci u datotekama moraju biti u najmanju ruku smisleni – kasnije ćemo videti da moraju zadovoljavati još uslove kako bi bili deo baze podataka. Znači baza podataka je u stvari neki skup datoteka. Naravno mora se napisati i program koji će upravljati tim podacima – potrebno je (najmanje) obezbediti upis novih izmenu i brisanje starih podataka. Softver koji omogućuje te operacije nazva se **sistem za upravljanje bezom podataka**. Taj softver i baza podataka zajedno čine jednu celinu koja se naziva **sistem baze podataka** ili ponekad samo **baza podataka**.

Podaci se u bazama podataka grupišu na razne načine u zavisnosti od tipa baze podataka. *Clipper* podržava takozvane **relacione** baze podataka. Kod takvih baze podataka podaci se predstavljaju u obliku **relacija** ili **tabela**. Na primer, spisak radnika u nekom preduzecu može se predstaviti relacijom

```
RADNIK (IME, PREZIME, RADNO_MESTO)
```

Teko, na primer, vaši **RADNIK (PERA, PERIĆ, MEHANIČAR)** jer je Pera Perić mehaničar u Joj firmi. Skup sv h trojki **(IME, PREZIME, RADNO_MESTO)** či ni relaciju **RADNIK**. I ona se vrlo pregledno može predstaviti u obliku sledeće tabela

IME	PREZIME	RADNO_MESTO
Pera	Perić	mehaničar
Mika	Mikić	operater
Nena	Nenić	službenik

Pri tome svaka vrsta date tabele predstavlja jednu trojku relacije **RADNIK**. Koristeći ovakvu terminologiju, bazu podataka možemo shvatiti ne samo **tabelę**

Primetimo kako ovakve tabele liče na strukturu standardnih sekvencijalnih datoteka: svaki slog datoteka ima određena polja i predstavlja jednu vrstu tabele. Tabele u bazi podataka se mogu ali ne moraju tako predstaviti. *dBASE* ma takvu strukturu- ali veliki broj relacionih baza podataka nema – sve tabele i relacije baze podataka mogu biti smeštene i u jednu datoteku.

Clipper koristi prvi pristup: svaka tabela baze podataka se predstavlja jednom datotekom sa ekstenzijom DBF koje ima polja koja odgovaraju kolonama tabele. Kad tabela sadrži mnmo polja uz DBF datoteku kreira se i datoteka sa ekstenzijom DBT u koju se upisuje tekst mamo polja. Znači jedne tabele može biti predstavljena jednom ili sa dve datoteke. DBF datoteka koju neki često (i sasvim neopravdano) nazivaju baza, zove se **datoteka baze podataka**, ali pošto je to malo predugo u ovom jeksiu ce se koristiti termin DBF datoteka ili samo datoteke. To pak nisu obične datoteke ali pošto se u *Clipper*-u relajivno rejko koriste standardne datoteke može se upotrebljavati ovaj termin bez opasnosti od zabune.

BAZA PODATAKA I CLIPPER

Clipper sadrži specijalni skup naredbi i funkcija koje omogućavaju rad sa tabelama, odnosno DBF datotekama. To su ne prvi pogled standardne sekvencijalne datoteka sa kojima upravljamo baš kao i u bilo kom drugom programskom jeziku. Na početku rada sa takvom datotekom potrebno je **otvoriti** je a na kraju je potrebno da se **zatvori**. Datoteka sa otvara naredbom **USE <naziv datoteke>** a zatvara naredbom **CLOSE <naziv datoteke>**. Sastavljanje je od slogova jednake duzine i strukture. Moguc je pristup ma kom slogu u datoteci naredbom **GO <broj sloga>** koja pozicionira pokazivač sloga datoteke na zadati slog. Kada se datoteka otvori pokazivač je na prvom slogu. Naredba **SKIP** služ za pomeranje pokazivača sloga na prvi naredni slog datoteke.

Već ovih nekoliko naredbi je dovoljno da se napiše jednostavan program

```
PROC Main
   USE Primer
   WHILE ! Eof()
      ? prezime, ime
      SKIP
   ENO DO
   CLOSE Primer
RETURN
```

Ovaj program ispisuje sve slogove datoteke **PRIMER**. Funkcija **Eof** ispituje da li je pokazivač sloga datoteke došao do kraja. U ovom primeru smo pretpostavili da svak slog datoteke **PRJMER** sadrži polja **IME** i **PREZIME** – kako se kreire strukjura sloga videćemo kasnije. Bitno je videti da su vrednosti u tekućem slogu dostupne bez ikakvog eksplicitnog učitavanje u memoriju – vrednosti polje tekućeg sloga automatski se učitavaju u memoriju posle svakog pomeranje pokazivača sloga. Postupak je potpuno automatski i programer ne treba da vodi brigu o tome. Preciznije rečano komunikacija izmedju datoteke i programa obavlje se pomoću specijalna memonjske oblasti koje se naziva **radna oblast ili radne zone**. Svakoj otvorenoj datoteci sa pridružuje jedna radna oblast u kojoj se pamte karakteristični podaci za tu datoteku: naziv, vrednosti polja tekućeg sloga, broj sloga itd. Vrlo je važno odmah po otvaranju svakoj datoteci dodeliti njenu radnu oblast. U programu se sve naredbe odnose na **tekuću** (aktivnu) radnu oblast, osim kada se eksplicitno ne naglasi oparacija u drugoj oblasti. Kada se datoteka otvare, ona se uvek vezuje za trenutno aktivnu radnu oblast. Kada program počne se izvršavanjem aktivna je **prva** radna oblast. Pre nego što se otvori naredna datoteke potrebno je aktivirati novu radnu oblast, u suprotnom se prethodno otvorena datoteka zatvare. Ovo se često zaboravlja, pa može ne pravti prilično velike probleme. Radna oblasti su numerisana brojevima od 1 do 250, a određivanje trenutno aktivna oblavlja naredba **SELECT <broj oblasti>**. Na primer

```
PROC Main
   // Aktivna je prva oblast
   USE Datoteka1

   // Aktivlra se druga oblast
   SELECT 2
   USE Datoteka2

   // Ponovo se aktivira prve oblast
   SEIECT 1

   // Oaloba"eju se obe radne oblasti
   CLOSE Datoteka1
   CLOSE Datoteka2
RETURN
```

Zapamlite da u jednoj radnoj oblasti istovremeno može biti otvorena samo jedna datoteka – eko poku šate da otvorite sledeću, pretnodna će se automatski zatvoriti.

ALIASI

Pošto je rad sa brojevima kao identifikatorima oblasti veoma nezgodan (šta li ono beše u oblasti 153?) preporučljivje je koristiti brv **alijase** (naki ih još nazivaju **pseudonimi**). Alias je simbolički naziv radna oblasti pomoću koga možemo pristupati oblasti bez korišćenje broje. Na primer, umesjo **SELECT 2** u pret hodnom primeru možemo pisati **SELECT Datoteka2**. Naravno, uslov za pristupanje radnoj oblasti pomoću alijasa je da je u njoj otvorena datoteka sa jim nazivom. Naime, kada se otvori datoteka u nekoj radnoj zoni ta radna zona automatski dobije alias koji je isti kao i naziv otvorena datoteke.

Pošto imena datoteke mogu biti prilično neprak tična, moguće je uvesjti naziv radna oblasji korišćenjem opcije **ALIAS** prilikom otvaranje datoteke. Na primer

```
SELECT 27
USE Prf01co3 ALIAS Podaci
```

Sad su naredbe **SELECT 27** i **SELECT Podaci** potpuno ravnopravne... ali je najbolje koristiti drugu varijantu. Pogrešno bi bilo pokušati sa **SELECT PRF01CO3** jer opcija **ALIAS** poništava naziv datoteka kao alias.

Početnici veoma često naprave totalnu zbrku sa aliasima i radnim zonama u programu, tako da novootvorene datoteke pri otvaranju zatvaraju prethodno otvorena i si. Pri radu se radnim zonama treba biti veoma pažljiv i najvažnije: doslediti. Treba strogo poštovati sledeća jednostavna pravila. Prvo datoteku otvarati na početku procedure/modula u kome je potrebna, a na kraju zatvarati. Ako na poštujete ovo pravilo, redovno ćete dobijati poruku da je nemoguće otvoriti toliko datoteka istovremeno. Treba izvršiti pažljivu selekciju i držati stalno otvorena samo neophodna datoteke. Pri otvaranju radnoj zoni obavezno treba dodeliti neko prepoznatljivo ime i nadalje datoteci u toj zoni pristupati isključivo pomoću jog imena. Otvaranje treba uvek vršiti u slobodnoj radnoj zoni, kako se ne bi greškom zatvorila neka potrebna datoteka. Ako se datoteke otvara sa opcijom **NEW**, automatski će se selektovati prva slobodna radna zone. Na primer, ako su našem programu potrebne datoteka **RADNICI**, **PLATE** i **PR1990**, program može ovako izgledati

```
PROC Main
   USE Radnici NEW
   USE Plate NEW
   USE Pr1990 NEW ALIAS Prihodi
   ...
   SELECT Plate
   // Aktivna je datoteka Plate

   SELECT Prihodi
   // Aktivna je datoteka Pr1990
   ...
   CLOSE Radnici
   CLOSE Plate
   CLOSE Prihodi
RETURN
```

Pojam "aktivna radna oblast" znači mnogo više nego da je moguć upis u selektovanu datoteku. Naime, već je rečeno da su vrednosti polja tekućeg sloga selektovane datoteke dostupna kao i sve "obične" promenljive. Na primer, ako datoteke radnici ima polja **IME** i **PREZIME**, a datoteka **PLATE** polja **BR_RADNIKA** i **IZNOS**, poste naredbe **SELECT Radnici** može se napisati **? IME, PREZIME** dok su polja **BR_RADNIKA** i **IZNOS** nedefinisani. Ako se izvrši naredba **SELECT Plate IME** i **PREZIME** postaju nedefinisani a može se pristupati poljima **BR_RADNIKA** i **IZNOS**. Tako se može pisati

```
IF Br_Radnika <= 100
   ? "PLATA + 20% ", Iznos * 1.2
END IF
```

Naravno polja mogu imati ista imena u nekoliko različitih datoteka, pošto je uvek aktivne jedna radna zona vidljiva su polja iz ta zone.

Pošto je vrlo često potrebno koristiti podatke iz nekoliko različitih zona (i unutar jednog izraze na pi-

mar) omogućen je pristup poljima iz drugih oblasti pomoću aliasa primenom operatora -> Na primer

```
Rnew    Plata->Bodovi [illegible]
? Nagrade->[illegible]
```

Ovde se uzima vrednost polja **BODOVI** iz radne zone **PLATA** i polja **IZNOS** zone **NAGRADE**. Ako se na navede alias uzima se polje iz tekuće radne zone. Ako takvo ne postoji traži se memorijska promenljiva sa tim imenom a ako ni ona ne postoji prijavljuje se *RUN TIME* greška.

Slično kao za polja alias se mogu koristiti i u kombinaciji sa izrazima. Naime, vrlo je čest slučaj da treba samo privremeno izračunati vrednost izraza u nekoj radnoj zoni različitoj od tekuće a rad treba nastaviti u tekućoj. Na primer

```
Z := X * Probe->(P1/Avg(P2))
```

U ovom izrazu se **Z** računa kao proizvod **X** i izraza **P1/Avg(P2)** pri čemu su polja **P1** i **P2** polja iz radne zone **PROBA**. Izraz koji se računa obavezno treba zatvoriti u zagrade. Ove (inače veoma korisna mogućnost) najčešće se koristi za proveru rezultata neke operacije u drugoj radnoj zoni. Tipičan primer je

```
Radnici->(Eof())
```

Rezultat ovog izraza je rezultat funkcije **Eof** (provera kraja datoteke) ali u radnoj zoni **RADNICI**.

PREGLED DATOTEKA

Već smo videli kako se vrše operacije sa celom datotekom: prvi način je pomoću kodnih blokova i funkcije **DBEval** a drugi pomoću "programskog" pomeranja pokazivača datoteke. Na primer sledeći primer ispisuje na ekranu celu datoteku **Radnici**

```
WHILE !Eof ()
   ? Ime, Prezime, Plata [illegible]
   [illegible]
[illegible]
```

U ovom jednostavnom primeru se ispisuju polja **IME**, **PREZIME** i zbir polja **PLATA** i **NAGRADA** za sva k slog datoteke **RADNICI**. Pošto se ovakve konstrukcije veoma često ponavljaju uvedena je posebna naredba koja odavlja sličnu operaciju

```
LIST Ime, Prezime, Plata + Nagrada [illegible]
```

Znači naredba **LIST** automatski počinje od prvog sloga i pomera pokazivač do kraja datoteke ispisujući pritom specificirane izraze za svaki slog.

Oznaka **ALL** označava da operaciju treba izvršiti za sve slogove datoteke. Znači, naredba **LIST** počinje od prvog sloga datoteke i obrađuje sve slogove do zadnjeg bez obzira na kom se slogu trenutno nalazi pokazivač sloga. Moguće je zadati i podskup slogova cele datoteke ako se umesto **ALL** navede **REST** podrazumeva se da obradu treba izvršiti za sve slogove počev od tekućeg do poslednjeg a opcija **NEXT n** specificira da obradu treba izvršiti za narednih **N** slogova. Na primer

```
[illegible] 81
  Ime, Prezime REST
```

ispisuje sve slogove počev od sloga broj 100 do kraja datoteke. Naredba **LIST** omogućuje i "precizinije" podešavanje skupa slogova na koji treba primeniti obradu: opcije **FOR** i **WHILE**. Opcijom **FOR** se zadaje uslov koji treba da zadovolji slog da bi bio obrađen. Na primer

```
LIST Ime, [illegible]
[illegible]
```

ispisuje sve slogove kod kojih je polje **PLATA** < **10000**. Opcija **WHILE** omogućuje da se obrada prekine kad bude zadovoljen određen uslov. Na primer

```
[illegible]
```

Za razliku od prethodne naredbe ova će ispisati sve slogove kod kojih je polje **PLATA** < **10000**, ali kad se pojavi prvi slog koji ne zadovoljava ovaj uslov **neće** se tražiti sledeći koji zadovoljava uslov nego će se ispisivanje prekinuti. **LIST** komanda dozvoljava još i opcije **TO PRINTER** i **TO FILE** kojima se izlaz usmerava na štampač ili u datoteku.

Postoji čitav niz naredbi koje obavljaju razne operacije sa slogovima aktivne datoteke koje rade po sličnim principima kao i naredba **LIST**. Na primer naredba **SUM** za sabiranje izraza na nivou cele datoteke, **COUNT** za prebrojavanje slogova koji zadovoljavaju određen uslov, **AVERAGE** za računanje prosečne vrednosti izraza itd. Na primer sabiranje svih plata za celu datoteku radnika može se obaviti ovako

```
SUM Plata * _Faktor  Nagrada
TO SumaPlata AL[illegible]
```

Sve napomene za dodatne opcije važe i kod ovih naredbi.

Interesantno (i korisno) je da su skoro sve ove naredbe realizovane preko već pomenute funkcije **DBEval** koja omogućuje izvršavanje kodnog bloka za zadati skup slogova u datoteci. Korišćenjem ove funkcije možete sami kreirati svoje naredbe koje će obavljati "makro" operacije sa celom datotekom ili njenim delovima.

KREIRANJE DATOTEKA

Pre nego što se počne sa korišćenjem **DBF** datoteke mora se definisati njena struktura: koja sve polja ulaze u slog, tipovi polja, maksimalne vrednosti itd. Kreiranje DBF datoteka se može obaviti na nekoliko sličnih načina. Jedan od njih je korišćenjem funkcija **DBCreate**. Princip je vrlo jednostavan: prvo se kreira niz u kome se navode nazivi polja (to su standardni identifikatori) koje će sadržati slog datoteke i razni dodatni atributi: maksimalna dužina,... Zatim se poziva funkcija **DBCreate** koja na osnovu niza kreira traženu datoteku. Pri zadavanju polja moraju se koristiti samo tipovi koje *Clipper* podržava, a to su:

Numeričko polje može sadržati broj, slično kao i promenljive ovog tipa. Pri kreiranju numeričkog polja treba zadati kolika će biti maksimalna širina celog zapisa i širina decimalnog dela. Na primer broj 12345.67 ima ukupnu širinu 8 (računa se i tačka) a širina decimalnog dela je 2. Oznaka za numeričko polje je **N**.

String je niz ASCII simbola. Pri kreiranju treba zadati maksimalnu dužinu stringa. Oznaka za string je **C**.

Datumsko polje može sadržati datum. Pošto je širina datuma uvek ista, pri kreiranju ne treba zadavati nikakve dodatne podatke. Oznaka za datumsko polje je **D**.

Logičko polje može sadržati vrednost **T** ili **F**. I za ovo polje ne treba zadavati dodatne atribute. Oznaka za logičko polje je **L**.

Memo polje je specijalan slučaj stringa. Naime ponekad je teško odrediti maksimalni broj simbola koji je potreban za upis podataka. Tada je besmisleno kreirati polje tipa string neke ogromne veličine jer će veći deo tog prostora ostati neiskorišćena. Memo polja su mnogo jednostavnije rešenje. U slogu DBF datoteke u kojoj je definisano memo polje ostavi se prostor za jedan pokazivač a kreira se posebna DBT datoteka u koju se smeštaju eventualno zadata memo polja. Na ovaj način se štedi prostor u DBF datoteci a odgovarajuće memo polje u DBT datoteci može biti proizvoljno dugo. Ipak postoji ograničenje od 64K na maksimalnu veličinu memo polja, što se u praksi retko dostiže. Memo polje se pri zadavanju obeležava sa **M**.

Podaci za jedno polje se zadaju u obliku niza od 4 elementa: naziv polja, tip, ukupna širina i širina decimalnog dela. Na primer niz **{ "PLATA", "N", 10, 2}** specificira numeričko polje **PLATA** ukupne širine 10 sa dve decimale. Argumenti za funkciju **DBCreate** su ime DBF datoteke koju treba kreirati (proizvoljan DOS naziv datoteke) i **niz** specifikacije polja u stvari niz čiji su elementi nizovi. Na primer

```
DefArray :=
{
   {"IME", [illegible]
   {"BROJ", [illegible]
   {[illegible]
}
DBCreate ("ADRESAR", DefArray)
```

U ovom primeru se prvo kreira niz specifikacija **DefArray** koji sadrži opise polja **IME** tipa string maksimalne dužine 30, numeričkog polja **BROJ** širine 10 sa dve decimale i datumskog polja **DATUM**. Primetite da se **uvek** zadaje niz od 4 elementa, mada u nekom slučajevima pojedini argumenti nemaju značenje (šta bi značio broj decimala kod stringa?). Zatim se poziva funkcija **DBCreate** koja kreira DBF datoteku **ADRESAR**.

Izmena strukture DBF datoteka nije baš jednostavna operacija i ne postoji funkcija koje samostalno obavlja tu operaciju. Ako želite da promenite strukturu već kreirane datoteke možete koristiti program *DBU* koji će biti opisan u nastavku teksta. Naravno može se napisati i procedura koja obavlja tu operaciju.

AŽURIRANJE DATOTEKA BAZE PODATAKA

Bez obzira šta čuvamo u **DBF** datoteci naš softver treba da realizuje najmanje tri operacije nad podacima u toj datoteci: dodavanje, brisanje i izmena podataka.

Naredba **APPEND BLANK** dodaje prazan slog na kraj aktivne datoteke. Pokazivač tekućeg sloga se automatski pomera na taj slog. Upis podataka u slog vrši se naredbom **REPLACE**. Sledeći primer pokazuje kako se dodaje novi slog u datoteku **RADNICI**

```
SELECT Radni[illegible]
APPEND BLANK
REPLACE Ime W[illegible]
"[illegible]"
```

Vrlo česta greška početnika je da ne upisuju podatke naredbom **REPLACE** već običnom naredbom dodeljivanja, što nikako ne treba raditi. Naime naredba **Ime := "Žika"** dodeljuje string **"Žika" memorijskoj** promenljivoj i ne upisuje u datoteku. Da stvar bude još komplikovanija ova naredba će upisati string na pravo mesto ali samo ako je promenljiva **Ime** deklarisana kao polje naredbom **FIELD**. Može vam se učiniti da je bolje sva polja i deklarisati ovako a zatim a njima postupati kao sa običnim promenljivim tj menjati im vrednost naredbom := ali ipak nije baš tako. *Clipper* ima veoma slabu kontrolu tipova promenljivih pa će vam se pre ili kasnije desiti da zaboravite da definišete neko polje a tada neće biti prijavljena nikakva greška, još gore program će naoko raditi korektno a na pronalaženje ovakve greške možete izgubiti sate i sate. Zato je bolje svaki upis u datoteku izvršiti naredbom **REPLACE**.

Izmena podataka se odvija takođe naredbom **REPLACE**. Pri izmene potrebno je pomeriti pokazivač sloga na slog koji želimo da izmenimo naredbom **GO**. Na primer ako želimo da izmenimo slog broj 20 pre odgovarajuće naredbe **REPLACE** izvršićemo **GO 20** ili neku drugu naredbu koja će pomeriti pokazivač na odgovarajuće mesto.

Kada treba izbaciti slog iz datoteke moraju se fizički iskopirati svi slogovi počev od prvog suseda sloga koji se briše do kraja datoteke. Pošto DBF datoteke mogu biti veoma velike ova operacija može prilično potrajati. Ako se uzme u obzir da treba pri brisanju sloga ažurirati adrese u indeksnoj datoteci i adrese memo polja u DBT datoteci dobija se vreme koje može biti **veoma** značajno. Zato se brisanje sloga odvija u dve faze. Prvo se označe svi slogovi koje treba obrisati naredbom **DELETE**. Posle toga se poziva naredba **PACK** koja briše sve označene slogove iz datoteke. Ako se u međuvremenu predomislite sa slogova označenih za brisanje možete "skinuti" oznaku za brisanje naredbom **RECALL**. Na primer ako treba izbrisati slogove broj 10, 120 i 500

```
GO 10 ; DELETE
GO 120 ; DELETE
GO 500 ; DELETE
PACK
```

Funkcija **Deleted()** vraća logičku vrednost koja označava da li je tekući slog označen za brisanje. Na primer

```
DELETE
? Deleted()  // .T.
RECALL
? Deleted()  // .F.
```

Ovaj mehanizam vam omogućava da korisnika poštedite suvišnog čekanja: obeležavanje slogova za brisanje je trenutna operacija pa tokom celog programa možete omogućiti slobodno označavanje nepotrebnih slogova a fizičko brisanje se može uraditi tek na kraju programa ili eksplicitnom naredbom korisnika. Ovo se pokazuje kao veoma pogodno jer ostavlja

korisniku mogućnost da se predomisli i "vrati" neki slog koji je greškom izbrisan – naravno pre izvršenja naredbe **PACK**.

PRETRAZIVANJE I IZDVAJANJE

Kada se DBF datoteke "napune" podacima, potrebno je korisniku na neki način omogućiti da od te gomile podataka dobije baš one podatke koji su mu potrebni. Taj zadatak mora da obavi naš *Clipper* program koji kontroliše bazu podataka. U principu se to može obaviti na dva načina: **traženjem** podatka koji zadovoljava neki uslov ili **izdvajanjem** svih podataka koji zadovoljavaju uslov.

Traženje podatka koji zadovoljava neki uslov obavljamo naredbom **LOCATE**. Ova naredba polazi od prvog sloga datoteke i redom izračunava zadati logički izraz za svaki slog datoteke, sve dok ne naiđe na slog koji zadovoljava uslov ili dok ne stigne do kraja datoteke. Ako je slog pronađen, pokazivač sloga se nalazi na prvom slogu koji zadovoljava uslov, a funkcija **Found()** vraća vrednost .T.. Ako slog nije pronađen, pokazivač se nalazi iza zadnjeg sloga datoteke, a funkcija **Found()** ima vrednost .F.. Na primer, traženje sloga koji zadovoljava neki uslov u programu obično izgleda ovako:

```
LOCATE FOR Ime == "Petar" .AND. Grad
== "Beograd"
IF Found()
   // Slog je pronadjen
ELSE
   // Ne postoji slog za zadati uslov
END IF
```

Vrlo često postoji više slogova koji zadovoljavaju zadati uslov. Naredba **LOCATE** pronalazi prvi slog koji zadovoljava uslov. Ako je potrebno naći i ostale slogove, koristi se naredba **CONTINUE**. Ona nastavlja započeto traganje počev od tekućeg sloga, pod istim uslovima kao i naredba **LOCATE**. Na primer:

```
LOCATE FOR BrLK == 72815
IF Found()
   CONTINUE
   IF Found()
      ? "Postoji i drugi slog koji za-
dovoljava uslov"
   ELSE
      ? "Nema drugog sloga!"
   END IF
ELSE
   ? "Nema sloga koji zadovoljava u-
slov!"
END IF
```

Vrlo često treba naći određen broj slogova, na primer, prvih nekoliko koji zadovoljavaju uslov ili sve koji zadovoljavaju uslov. Velika je greška ako se za to koristi naredba **LOCATE** u ciklusu – ova naredba uvek traži slog od početka datoteke, pa će uvek "naći" prvi slog koji zadovoljava uslov. U ovakvim slučajevima samo prvi slog treba tražiti naredbom **LOCATE**, a sve ostale naredbom **CONTINUE**.

Osim traženja sloga koji zadovoljava zadati uslov, mogu se izdvojiti svi slogovi koji zadovoljavaju uslov. Za to služi naredba **SET FILTER**. Korišćenjem te naredbe zadaje se logički uslov koji u aktivnoj datoteci prividno izdvaja samo one slogove koji zadovoljavaju zadati uslov. Važno je znati da je ovo sužavanje samo prividno i da se datoteka "smanjuje" računanjem izraza za svaki slog. Ova operacija može biti veoma osetljiva iz više razloga. Osnovni razlog je što se pri svim kasnijim operacijama računa iznova logički izraz za svaki slog kome treba pristupiti, što može biti veoma sporo. Zato upotrebu ove naredbe treba što je moguće više izbegavati. U sledećem primeru je prikazano izdvajanje svih slogova u datoteci **FAKTURE** koji sadrže podatke stanje od nekog datuma.

```
SET FILTER TO F_Dat >= Date() - 30
// Filter se ukljucuje pomeranjem po-
kazivaca na pocetak datoteke
GO TOP
LIST F_Dat, F_Broj, F_Naziv ALL
```

Filter se isključuje ili specificiranjem novog filtera naredbom **SET FILTER** ili naredbom **SET FILTER TO** bez argumenata koja ukida aktivan filter.

Često je uslov za izdvajanje veoma složen i njegovo računanje može prilično trajati, što značajno usporava rad. Međutim, treba imati u vidu da glavninu vremena u opštem slučaju troši rad sa datotekom, a ne računanje. Postavlja se pitanje, zašto Clipper uopšte uvek iznova računa izraz za svaki slog datoteke, tj. zašto ne izračuna izraz na početku za celu datoteku i tako interno označi slogove koji zadovoljavaju uslov, a zatim filter postavi na tako označene slogove. Problem je u tome što uslov može biti vezan na neki način za slog, na primer, preko broja sloga ili neke vrednosti u slogu. U tom slučaju je komplikovano analizirati izraz i računati ga samo kad je potrebno. Međutim, to nas ne sprečava da mi sami nešto ne preduzmemo. Prvo, uvek kad je moguće treba upotrebljavati kodni blok umesto standardnog uslova, jer se ovaj mnogo brže izvršava. Drugo, u slučaju složenog uslova može se uvesti dodatno logičko polje za svaki slog koje će označavati da li je slog "vidljiv", izračunati uslov za svaki slog, a zatim postaviti filter na vrednost tog polja.

INDEKSIRANJE

U suštini se vrlo retko traži slog po nekom komplikovanijem izrazu, u većini slučajeva je potrebno naći slog po nekoj karakterističnoj vrednosti koja je jedinstvena za svaki slog (tzv. **ključ sloga**). U ovakvim (vrlo čestim) slučajevima nikako ne treba koristiti naredbu **LOCATE** za traženje sloga – ova naredba pretražuje redom slogove i to može veoma dugo trajati. Za ovakve slučajeve uvode se **indeksi**.

Zadržaćemo se na trenutak na problemu koji se indeksom rešava. Problem nalaženja elementa u nizu brojeva koji ima određenu vrednost veoma je star. Naredba **LOCATE** realizuje takozvano sekvencijalno pretraživanje, koje je najsporije od svih. Tu se redom ispituje svaki element niza sve dok se ne pronađe odgovarajući. Znači, ako je traženi element poslednji u nizu, a niz ima N elemenata, biće izvršeno N ispitivanja. Veliki napredak je ako su vrednosti u nizu sortirane, tada se može primeniti binarno pretraživanje. Uzme se element iz sredine niza i uporedi njegova vrednost sa traženom. Ako je niz rastući, a vrednost koju tražimo veća od izdvojenog elementa, to znači da se element koji tražimo nalazi sigurno desno od izdvojenog elementa. Time smo u startu eliminisali N/2 poređenja. Postupak se analogno nastavlja za ostatak niza. Ovde se dobija na brzini, ali se rastuća vrednost elemenata niza uvek mora održati, pa je zato ubacivanje novih i brisanje starih elemenata niza otežano. Tako se na jednoj (manje važnoj) strani gubi, a na drugoj (mnogo važnijoj) strani dobija u vremenu obrade.

Ovakvo pretraživanje nikako nije najbrže, mnogo brži algoritam je pretraživanje korišćenjem B-stabla, u kome se traženi element dobija sa još manje operacija. Naravno, B-stablo je još komplikovanija struktura od sortiranog niza, pa je ubacivanje i brisanje elemenata znatno komplikovanije. Pošto se podaci u Clipper-u čuvaju u DBF datotekama, potrebno je slogove ovih datoteka "poređati" u ovu ne baš jednostavnu strukturu. Naravno, fizičko pomeranje slogova ne dolazi u obzir – manipulacija sa datotekom od nekih 10MB nije baš elegantna operacija! Primenjen je potpuno drugi pristup: kreira se posebna datoteka u kojoj se čuva stablo potrebne vrednosti i pokazivači na odgovarajuće slogove u matičnoj DBF datoteci. Tako se traženje sloga izvodi na sledeći način: prvo se u indeksnoj datoteci pronađe odgovarajući element, a zatim se pomoću pridruženog pokazivača pristupa slogu u DBF datoteci. Ovaj postupak je veoma brz i omogućava skoro trenutno pronalaženje vrednosti u datoteci.

Da bi se indeksi koristili, potrebno je prvo kreirati indeksnu datoteku. Ona ima ekstenziju NTX i kreira se naredbom **INDEX ON**. Na primer, sledeći primer indeksira datoteku **RADNICI** po matičnom broju radnika:

```
USE Radnici
INDEX ON R_Matbr TO I_Rad
```

Pri tom je **R_MATBR** numeričko polje u datoteci **Radnici**. Posle ove naredbe kreira se datoteka **I_RAD.NTX** koja omogućava pretraživanje datoteke **RADNICI** po matičnom broju. Za to služi naredba **SEEK**. Na primer, traženje radnika koji ima matični broj 12563 može se izvršiti naredbom **SEEK 12563**. Slično kao za naredbu **LOCATE**, ako slog sa tom vrednošću polja **R_MATBR** postoji, pokazivač sloga će biti na tom slogu, a funkcija **Found()** imaće vrednost .T..

Naredba **INDEX ON** se u opštem slučaju vrši samo jednom, sve kasnije izmene na DBF datoteci Clipper sistem automatski prenosi na odgovarajuću indeksnu datoteku. Na primer, brisanje, izmena i dodavanje novih slogova automatski se odražava i na NTX datoteku. Postupak je potpuno automatski i programer ne treba o tome da vodi računa. Jedini uslov je da uvek kad se vrše izmene na odgovarajućoj DBF datoteci budu otvorene i odgovarajuće indeksne datoteke. Kada se jednom kreira indeksna datoteka, sledeći put nije potrebno da se kreira ponovo, dovoljno je pri otvaranju otvoriti i indeksne datoteke.

```
USE Radnici INDEX I_Rad
```

Za svaku DBF datoteku može se kreirati više indeksa i po potrebi aktivirati jedan od njih. U jednom trenutku može biti aktivan jedan indeks. Koji će indeks biti aktivan određuje se naredbom **SET ORDER**. U sledećem primeru se kreiraju tri indeksne datoteke.

```
USE Radnici
INDEX ON R_Matbr TO Radnici1
INDEX ON R_Ime TO Radnici2
INDEX ON R_Staz TO Radnici3
```

Posle izvršenja ovih naredbi, **datoteka RADNICI** se može otvoriti na sledeći način:

```
USE Radnici INDEX Radnici1, Radnici2,
Radnici3
```

Posle ovoga je aktivan index iz datoteke **RADNICI1**, a to je **R_MATBR**. Bez obzira što preostala dva indeksa nisu trenutno aktivna, odgovarajuće **NTX** datoteke će biti korektno ažurirane sa svakom promenom DBF datoteke. Ako se izvrši naredba **SET ORDER TO 2**, aktiviraće se indeks iz datoteke **RADNICI2**, uopšte, **SET ORDER TO N** aktivira N-ti indeks. **SET ORDER TO 0** isključuje sve indekse.

Kao izraz za indeksiranje može se upotrebiti proizvoljan izraz tipa string, broj, datum i logički tip.

Indeksiranjem se slogovi na neki način i sortiraju, sve naredbe "vide" DBF datoteku kao da je sortirana po trenutno aktivnom indeksu. Na primer, ako u datoteku **PROBA** koja ima numeričko polje **P1** upišemo slogove 2, 3, 5, 4, 1 tim redom, naredba **LIST** ispisaće slogove onim redosledom kako su i uneti. Ako se datoteka indeksira po polju **P1**, datoteka će se ponašati kao da je slog 1 prvi upisan, itd. Naredba **LIST** ispisaće brojeve u sledećem redosledu: 1, 2, 3, 4, 5. Važno je zapamtiti da datoteka nije fizički menjana, tako da je broj slogova zadržan. Na sledećoj tabeli vide se rezultati.

PRE INDEKSIRANJA		POSLE	
VREDNOST	RECNO()	VREDNOST	RECNO()
2	1	1	5
3	2	2	1
5	3	3	2
4	4	4	4
1	5	5	3

Brojevi se indeksiraju u rastućem poretku. Ako je potrebno indeksiranje u opadajućem poretku, koristi se funkcija **Descend()** koja prihvata izraz bilo kog tipa i vraća njegov komplement. Na primer, ako je potrebno indeksirati po polju **P1** u opadajućem poretku, treba izvršiti naredbu **INDEX ON Descend(P1) TO I_Probe**.

Indeksiranje po string izrazima je posebno interesantno, jer je vrlo često potrebno prikazati podatke po abecednom redosledu. Clipper indeksira podatke koristeći ASCII tabelu, u kojoj, na žalost, nema naših simbola **ŠĆČŽĐ**, tako da sortirani podaci neće biti baš idealno poređani, ali bolje išta nego ništa. Za rešenje ovog problema pogledajte stranu "Računare".

Indeksiranje po datumskom izrazu je dosta često, ali da bi se izvršilo korektno sortiranje takvih podataka (datumi su poređani po hronološkom redosledu), potrebno je indeksirati koristeći funkciju **DtoS()** koja datum transformiše u pogodan izraz za indeksiranje.

Jedan od problema koji često muče početnike je kako korišćenjem **SEEK** naredbe naći ostale slogove koji imaju istu vrednost indeksa. Na primer, datoteka faktura je indeksirana po datumu (**DtoS(F_Dat)**) i treba obraditi sve fakture koje su došle jednog dana. Naredba **SEEK DtoS(dat_obr)** će naći prvu takvu, ali kako naći sledeću? Koristiti naredbu **LOCATE**, a zatim **CONTINUE** ovde nema mnogo smisla kad su na raspolaganju mnogo brže varijante. Treba uočiti da je

sledeći slog koji zadovoljava uslov sada odmah iza prvog i tako dalje sve do poslednjeg. U našem primeru program ća izgledati otprilike ovako

```
SEEK DtoS(dat_obr)
WHILE dat_obr == F_Dat
    // Obrada
  SKIP
END DO
```

VIŠESTRUKE RELACIJE

U "Računarima" je već bilo reči o povezivanju više DBF datoteka, zato ćemo ponoviti samo osnovne stvari.

Svaka DBF datoteka u bazi podataka predstavlja jedan realni objekat. Često se javlja slučaj da su podaci u dve ili više datoteka na neki način povezani. Na primer, datoteka pisaca i datoteka knjiga može se povezati vezom koja se može nazvati **napisao je** – svaki autor je napisao jedno ili više dela. Naravno, ova veza se može posmatrati i sa druge strane: svaku knjigu je **napisao** neki autor. Pretpostavimo da imamo datoteku **KNJIGE** sa poljima **Naziv** (naziv dela) i **AutNo** (identifikacioni broj autora dela) i datoteku **AUTORI** sa poljima **AutID** (identifikacioni broj autora) i **ImePrez** (ime i prezime autora), indeksiranu po polju **AutID**. Kako sad ispisati sve parove **(NAZIV_DELA, AUTOR_DELA)**? Najprirodnije rešenje je da se za svaki slog u datoteci **KNJIGE** pronađe odgovarajući slog u datoteci **AUTORI** naredbom **SEEK**. Na primer

```
USE Autori INDEX Autori NEW
USE Knjige NEW
WHILE !Eof()
   ? Naziv
   SELECT Autori
   SEEK Knjige->AutNo
   ?? ImePrez
   SELECT Knjige
   SKIP
END DO
```

Ovaj postupak se može potpuno automatizovati korišćenjem naredbe **SET RELATION**. Ona povezuje dve radne oblasti vrlo slično prethodnom segmentu programa: uvek kada se pomeri pokazivač sloga u prvoj datoteci biće izvršena naredba **SEEK** u drugoj oblasti sa odgovarajućim argumentom. Tako će biti pronađen odgovarajući slog u drugoj datoteci. Za prethodni primer treba izvršiti sledeću naredbu

```
SET RELATION TO AutNo INTO Autori
```

Ova naredba označava da treba po pomeranju pokazivača u tekućoj datoteci izvršiti u oblasti sa alijasom **Autori SEEK** naredbu sa argumentom **AutNo** iz tekuće oblasti **Knjige**. Na taj način se prethodni primer može napisati kao

```
USE Autori INDEX Autori NEW
USE Knjige NEW
SET RELATION TO AutNo INTO Autori
LIST Naziv, Autori->ImePrez ALL
```

Jedna datoteka (bolje rečeno radna zona) može se povezati istovremeno sa više radnih oblasti. Na primer, ako postoji datoteka izdavača knjiga, može se napisati

```
USE Autori INDEX Autori NEW
USE Izdavaci INDEX Izdavaci NEW
USE Knjige NEW
SET RELATION TO AutNo INTO Autori,
IzdNo INTO Izdavaci
LIST Naziv, Autori->ImePrez,
Izdavaci->Naziv ALL
```

Ako se posle jedne **SET RELATION** naredbe navede sledeća u istoj radnoj oblasti, prethodna će biti poništena, što treba imati na umu. Ako se tekuća oblast višestruko povezuje, treba koristiti jednu naredbu kao u prethodnom primeru ili posebnu opciju **ADDITIVE**. Kada se navede ova opcija relacija se dodaje na prethodno aktivne. Tako je prethodni primer ekvivalentan sa

```
SET RELATION TO AutNo INTO Autori
SET RELATION TO IzdNo INTO Izdavaci
ADDITIVE
```

Ne postoji nikakva prepreka da se datoteke dalje lančano povezuju. Jedino treba voditi računa da se ne napravi ciklus – ako povežete jednu zonu direktno ili indirektno sa samom sobom program će se zaglaviti.

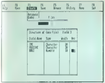
U prethodnom primeru smo zanemarili razna činjenice, između ostalog i slučaj kad u drugoj oblasti ne postoji odgovarajući slog. Tada će naredba **SEEK** pozicionirati pokazivač sloga iza kraja datoteke, a funkcija **Found()** u oblasti **AUTORI** imaće vrednost F. Da bi se predupredili ovakvi slučajevi treba pre štampanja imena autora proveriti postojanje odgovarajućeg sloga izrazom **Autori->(Found())**. Potpuno pogrešno bi bilo koristiti samo **Found()** jer bi se to odnosilo na tekuću radnu oblast u kojoj je to potpuno različita vrednost.

RAZVOJ PROGRAMA U CLIPPER-U

Kao što je već rečeno, *Clipper* je sistem za razvoj kompletnih aplikacija. Osim prevodioca i linkera, ceo paket sadrži čitav niz korisnih alatki koji vam mogu značajno pomoći. Te dodatne alatke početnici a i iskusniji programeri dosta retko koriste, najverovatnije zbog neinformisanosti. Naredni tekst sadrži kratke opise nekih od tih alatki, kao i praktikum za neke stvari koje se mogu javiti u svakodnevnom radu.

OPERACIJE SA DBF DATOTEKAMA

U prethodnom tekstu smo videli kako se DBF datoteka može kreirati programski, korišćenjem funkcije **DBCreate**. Međutim, baza podataka može biti veoma komplikovan sistem datoteka i nema mnogo svrhe da program koji radi sa bazom podataka kreira potrebne datoteke, osim u nekim specijalnim slučajevima. Pošto se kreiranje datoteka obavlja samo jednom, sasvim je dovoljno da uz program isporučite prazne datoteke. Postavlja se pitanje kako najlakše kreirati DBF datoteku. *Clipper* sadrži program za rad sa DBF datotekama *DBU* koji će vam pomoći da jednostavne operacije sa datotekama uradite bez pisanja programa.



Radni ekran programa *DBU* sa osnovnim opcijama koje se mogu birati pritiskom na funkcijske tastere F1-F8

Posle startovanja, *DBU* ispisuje radni ekran sa osnovnim opcijama koje se mogu birati pritiskom na funkcijske tastere F1-F8 (slika 2). Kreiranje nove datoteke se obavlja na sledeći način: prvo se izabere opcija **Create Database** (F3), a zatim se popune podaci o strukturi datoteke. U prvoj koloni se kuca naziv polja, u drugoj tip, a u trećoj i četvrtoj dužina celog i decimalnog dela. Za neke tipove podataka nije potrebno navoditi dužinu. Novi podatak o polju se upisuje tako što se pritisne strelica na dole kad je kursor na poslednjem podatku. Po definisanju strukture treba fizički kreirati datoteku. To se radi izborom opcije F4 (**Save Struct**). Pre snimanja treba uneti ime kreirane datoteke i potvrditi izbor.

Za komunikaciju se konsolkom *DBU* koristi tzv. dijalog boksove sa veoma fleksibilnim kretanjem kursora. Na primer, kada već unesete ime datoteke jednostavnim pritiskom na strelicu gore možete ponovo da promenite taj podatak ili pomeranjem u desno da izaberete neku postojeću datoteku sa spiska.

Kada se datoteka kreira, automatski se otvara u prvoj slobodnoj radnoj oblasti. Pošto pre ove operacije nije otvorena ni jedna datoteka, datoteka koju smo kreirali otvorena je u prvoj radnoj oblasti. To je označeno nazivom datoteke u sredini ekrana ispod natpisa **Files**. Ispod ovog natpisa stoje natpisi **Indexes** i **Fields** koji se odnose na tekuću datoteku. Za naš primer ne postoji indeks jer je datoteka upravo formirana. Kreiranje indekse se izvodi izborom opcije **Create Index** (F3) iz menija. Po izboru ove opcije traži se da unesete naziv indeksne datoteke i izraz po kome treba indeksirati datoteku. Ovde treba napomenuti da izraz mora sadržati samo originalne *Clipper* funkcije koje su već ulinkovane u *DBU*, ako za indeksiranje koristite neku vlastitu funkciju, za to nećete moći da koristite *DBU*. Kada se datoteka indeksira, ispod natpisa **Indexes** stoji naziv indeksne datoteke koju smo upravo kreirali.

Ako nešto od onoga što smo kreirali treba menjati, dovoljno je kursorom izabrati željeni objekat (na primer za izmenu strukture naziv datoteke) i ponovo izabrati opciju **Create**.

Već kreirane datoteke se otvaraju opcijom **Open Database** (F2). Iz ponuđenog spiska datoteka treba zabrati naziv odgovarajuće (pritisak na slovo izvodi skok na datoteku čiji naziv počinje tim slovom) ili otkucati naziv u levom kraju dijalog boksa. Otvorena datoteka se dodeljuje prvoj sledećoj radnoj oblasti, njen naziv se pojavljuje pored naziva datoteke koju smo kreirali. Na ovaj način možemo otvoriti više DBF datoteka. Indeksne datoteke se analogno otvaraju. Pre otvaranja indeksne datoteke kursor treba pomeriti na kolonu u kojoj je odgovarajuća (već otvorena) DBF datoteka. Zatim se bira opcija **Open Index** (F2). Ako datoteka ima više indeksa, potpuno analogno se otvaraju i ostali.

Sledeća važna mogućnost koja se često koristi je tabelarni pregled otvorene datoteke. Prvo se izabere kolona sa odgovarajućom datotekom, a zatim se izabere opcija **Browse Database** (F5) iz glavnog menija. Posle ovoga se dobija tabelarni pregled izabrane datoteke u kome možemo slobodno dodavati, menjati, brisati podatke. Tekući slog se briše pritiskom na taster **<Delete>** čime se samo markira za brisanje – fizičko brisanje se izvodi izborom opcije **Pack** iz menija **Utility** (F6). Ovaj meni se može pozivati tek kad se izađe iz tabelarnog pregleda.

Jedna napomena: kada je datoteka prazna, zbog greške u programu, u tabelarnom pregledu ne treba odmah kucati podatke – program će se zaglaviti. Pre kucanja treba pritisnuti strelicu na dole čime se dodaje prazan slog.

Nabrojane mogućnosti su samo jedan deo od onoga šta može program *DBU*. Može se vršiti tabelarni pregled više datoteka, postavljanje relacija, filtera, brisanje svih slogova u datoteci i sl.

Iz programa se izlazi pritiskom na taster **<Esc>**.

CLIPPER PREVODILAC

Clipper ima sasvim standardnu koncepciju prevođenja i linkovanja programa. Njegov prevodilac **CLIPPER.EXE** spada u veoma brze, ali, na žalost, to je uglavnom zato što i nema previše posla oko prevođenja i optimizacije programa. Prevodilac se standardno poziva komandom

```
CLIPPER <naziv datoteke> <opcije>
```

Opcije omogućavaju da prevod uskladimo sa našim trenutnim potrebama – kad ispravljamo greške u programu, potrebno je da imamo informacije o broju linija izvornog koda u OBJ datotekama, a ako pravimo demo verziju to nam nije potrebno. *Clipper* prevodilac podržava sledeće "prekidače":

/B Uključivanje *debug* informacija u OBJ datoteku. Ova opcija je neophodna kada se koristi dibager.

/D – Definiše meta promenljivu za pretprocesor. Na primer, ako se navede opcija **/Dinstall** meta promenljiva **INSTALL** postaće definisana, pa će pretprocesorski uslov **#ifdef install** biti ispunjen.

/L – Isključuje brojeve linija izvornog koda iz OBJ datoteka. Ovi brojevi se koriste kod rada sa dibagerom i kod izveštaja o greškama. Prilikom razvoja programa redovno treba izostavljati ovu opciju, ali kad se pravi instalaciona verzija programa treba obavezno uključiti opciju /L jer se tako smanjuje EXE datoteka.

/N Ne podrazumeva se da datoteka ima slo ime kao i procedura u njoj. Ovu opciju obavezno upotre-

bljavajte. Naime, može se desiti da napišete program, sve potrebne module prevedete i linkujete a po startovanju program samo vrati prompt – to ja siguran znak da niste upotrebljavali ovu opciju pri prevođenju.

/P – Kreira se rezultat rada preprocesora kao datoteka sa ekstenzijom PPO. Ovo je korisna opcija ka da želite da vidite stvarni izgled prevoda ili kada ispravljate neke svoje preprocesorske direktive.

/W – Veoma korisna opcija – generiše upozorenje za sva nedeklarisana i sumnjive promenljive. Na primer, ako u sred programa napišete **Xper := 1**, a niste unapred deklarisali promenljivu **Xper**, automatski će biti kreirana promenljiva klase **PRIVATE**. Međutim šta ako ste greškom napisali **Xper** umesto **Xperc**? Tada ovo postaje veoma ješko u otkrivanju greška, jer program radi bez greške. Kada se prevod vrši opcijom **/W**, *Clipper* će prijaviti grešku za svaku promenljivu koja je upotrebljena a nije deklarisana. Jedina mana ove opcije je što morate eksplicitno navoditi klase za polja u radnoj zoni.

/Z – Isključuje optimizaciju računanja logičkih uslova. Na primer, ako treba izračunati **<Cond1> .AND. <Cond2>** i izračuna se da je **<Cond1>** netačno, nema svrhe računati i **<Cond2>** bez obzira na vrednost ceo izraz je netačan. Slično je i sa **OR** operatorom. Ove optimizacija je uključena ako se ne navede opcija **/Z**.

RTLINK

Negativno nasleđe koje je *Clipper Summer '87* ostavio svom potomku je nepoverenje u linker. Zaista, *PLINK86* je bio čudo od sporosti i neiskazivosti. O gromna većina programera sa mahom opredeljivala za naupoređivo brži *TLINK* ili ostatno brži *Microsoft* linker. Novi linker koji se isporučuje uz novi *Clipper* obavezno treba koristiti jer se radi o odličnom proizvodu koji rešava više problema, a najvažniji su nedostatak RAM memorije i veličina EXE datoteka.

Ako koristite *rTlink*, sasvim sigurno nećete imati problema sa nedostatkom memorije jer on program automatski deli na overleje koji će se kasnije učitavati po potrebi. Ovaj veliki problem starog *Clipper*-a rešen je u potpunosti.

rTlink se poziva na sledeći način:

```
rtlink file proba,grnaal,grane2 lib
nt2,cligpr
```

Iza reči **FILE** (može se skratiti samo na **F**) sledi spisak datoteke koje čine ceo program. Datoteka koje sadrži glavnu proceduru (**Main**) more biti navabena prva. Iza reči **LIB** sledi spisek biblioteka koje program koristi. U ovaj spisak treba udaciti sistemske biblioteke kao što su **CLIPPER LIB** i **EXTEND LIB**, ali da se to ne bi stalno kucalo, obavezno izvršite DOS komandu

```
SET LIB = <LibDir>
```

Gde je **<LibDir>** katalog na kome ste smestili *Clipper* biblioteka prilikom instalacije. Tako će *rTlink* automatski tražiti šta mu je potrebno od sistemskih biblioteka, a vi iza reči **LIB** navodite samo dodatne biblioteke. Ako ih nema (a najčešće je tako) reč **LIB** možete izostaviti.

rTlink dozvoljave tzv. slobodni format zapise što podrazumeva da se cele komandne linije može navesti sa proizvoljno razmaka i u više redove. Naravno ovo ima smisla koristiti samo ukoliko je spisak datoteke za linkovanje veliki. Tada se spisak može nastaviti u sledećem redu daz posebnog označavanja kraja prethodnog reda. Prvo treda otkucati samo *rTlink* (bez argumenata) i dobija se prompt => Na primer:

```
C:\> rtlink
.*TLink for Clipper  Dynamic Overlay
Linker/Pre-Linker Version 3.13
(C) Copyright Pocket Soft Inc., 1988-
1991.  All Rights Reserved

=> fi test, test1,
=> test2
=> lib mt2
=> r
```

Da bi se označio kraj unosa treba otkucati " " (taćka-zarez). Međutim veliki spisak datoteke se bolčno neće kuca u komandnoj liniji – ko bi to uvek iz početka ukucavao! Mnogo bolje rešenje je tzv. *script* datoteka. U toj datoteci se napišu svi argumenti za *rTlink* a kasnije se on poziva sa **RTLINK @<ScriptFile>** gde je **<ScriptFile>** naziv datoteke sa parametrima. Na taj način ušteđimo na kucanju velikog broja naziva datoteka – dovoljno je samo da unosimo promene u ovu datoteku.

rTlink ima veliki broj opcija pomoću kojih finije utičemo na razukat njegovog rada. Pomenućemo samo najinteresantnije. Svim opcijeme je zajedničko da se navode iza samog poziva *RTLink*-a. Na primer **rtlink /resident fi proba**,

Opcija **/DYNAMIC** služi za kreiranje dinamičkih overleja. *RTLink* će sva module koji ulaze u aplikaciju podeliti na overleje i tako smanjiti minimalnu RAM memoriju potrebnu za startovanje programe. Po startovanju aplikacije u memoriju učitava se samo osnovni overlej koji je stalno prisutan u memoriji (tzv. *root* overlej) a ostali se učitavaju po potrebi kada se pozove odgovarajuća procedure i funkcije. Pošto obično ne mogu stati svi moduli u memoriju pra učitavanja novih se izbacuju prethodno učitani. Da bi se izbeglo stalno učitavanje i brisanje procedura koji se često pozivaju koristi se **LRU** algoritam. Za svaki modul se vodi evidencije o broju poziva i u memoriji ostaju samo oni koji se najčešće koriste. To obezbeđuje da program u prvih nekoliko sekundi "uči" šta se češće a šta rade koristi i tada se oseća smanjenje brzine zbog učitavanja modula sa diska. Kasnije se učitavanje vrlo slabo primećuje i program radi sasvim normalno.

Ipak u nekim slučajevima je dobro isključiti dinamičke overleje. Na primer program je mali i postoji dovoljno RAM memorije – tada nema smisla usporavati program dinamičkim overlejima, pa je bolje navesti opciju koja ih isključuje **/RESIDENT**. Time će se program naći obrzati ali će zato zahtevati znatno veću memoriju za startovanje. Mogućs je da ga uopšte i nećete moći startovati.

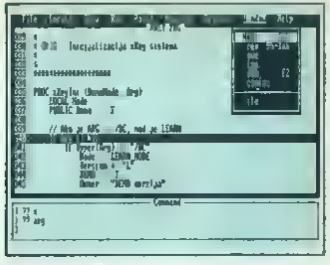
Opcija **/INCREMENTAL** uključuje inkrementalno linkovanje. To je postupak kojim se štedi na vremenu linkovanje u slučaju velikih aplikacije. Naime, obično se pre linkovanja vrši izmena samo u jednom modulu pa zašto onda menjati EXE datoteku celog programa kad je dovoljno izmeniti samo jedan njen delić? Problem je što je taj "delić" negde u sredim i posle izmene može postati veći od prathodnog i tada je jedino rešenje da se ubaci na kraj aplikacije. Taj postupak je i primenjen u inkrementalnom linkovanju u posebnoj datoteci se vode podaci o modulima koji čine ceo program i kad se vrši ponovno linkovanje menjaju se samo moduli čije su OBJ datoteke menjane. Tekvi moduli se dodaju na kraj EXE datoteke čime se veličina EXE datoteke sveki put povećava. Kada veličina nastane iznad nekog zadatog nivoa (može se zadati u procentima posle reči **INCREMENTAL**) ponovo se kreira cela EXE datoteka. Ove opcija služi za razvoj programa i njen efekat se oseća samo u slučaju veoma velikih programa. Kada je program manji vreme koje se troši na proveru zavisnosti može biti i veće od "normalnog" linkovanja. Na žalost ove opcija linkera ima grešku tako da ovako linkovan program vrlo često javlja neke nepostojeće greške ili se jednostavno zaglavi, tako da se primenom ove opcije treba biti vrlo obazriv i najbolje je sačekati novu verziju *RTLink*-a.

Mnogo korsnije opcija je **/PLL:<PllLib>** koje omogućava korišćenje tzv. daljenih biblioteka. Naime poznato je da *Clipper* u EXE datoteci sadrži ogroman deo sistemskih bibi oteka ze korisnički interfejs. prostup bazi i sl. Pošto na disku odično postoji više *Clipper* programa prostor koji oni zauzimaju može biti veoma znacajan. Zašto onda na napraviti jednu zajedničku datoteku sa tim sistemskim proceduramа, koju će posebni *Clipper* programi pozivati po potrebi? Upravo o vakva ideje je iskorišćena za tzv. **PLL** biblioteke. Posebnom opcijom *RTLink*-a može se kreirati spacijalna biblioteka sa modulima koje koriste svi *Clipper* programi na disku. Zatim se b programi linkuju sa opcijom **/PLL:<PllLib>** gde je **<PllLib>** ime kreirane PLL biblioteke. Tada *RTLink* u EXE datoteku uključuje sve osim procedura iz PLL b biblioteke čime se dobija EXE datoteke smanjena ze oko 150 200Kl. Naravno ovakav program se nešto sporije izvršava i za izvršavanje je potrebno da na disku postoji odgovarajuća PLL biblioteka što je sasvim zadovoljavajuće za vreme razvoje programe. kada program treba installrab na drugi disk treba ga linkovati dez PLL opcija. Uzgredni efekat ove opcije je mnogo brži rad linkere – aplikacije se kreira 2-3 puta brže. Uz *Clipper* se isporučuje PLL bi blioteka **BASE50** koja sadrži sistemske module *Clipper*-a. PLL biblioteki možete i vi kreirati od procedura koja su zajedničke za sve vaše programe i tako dra stično smanjiti zauzeća prostora na disku i skratiti vre me linkovanje.

Treba pomenuti i zgodnu opciju **/HELP** posle ko ja *RTLink* ispisuje obaveštenja o raznim opcijama.

DEBUGGER

U razvoj programa se nimalo zanemarljivim procentom ulazi i ispravljanje grešaka. Porad koncentraci je prilikom pisanje programe (ovo je prsventivel) još jedno srsdstvo je nazamenljivo u pronalaženju i ispravljanju učinjenih grešaka dibager. Njegovim korišćenjem možete da izvršavate program naredbu po naredbu, do određenog mesta ili dok se određeni uslov ne ispuni. Da vidite kako se menjaju pojedina promenljive i sl. Dibager koji se isporučuje uz *Clipper* je prilično prosečan – nove generacije dibagera imaju mnogo veća mogućnosti ali i ovaj je u sušini veome korisna i upotrebljive alatka. Ovde će biti nabrojana samo njegove osnovne mogućnosti.



Radni ekran dibagera odakle se mogu birati razne komande

Da bi se dibager koristio, sve datoteke koje želimo da "posmatramo" treba prevoditi sa opcijom **/B** koja saopštava *Clipper* prevodiocu da u OBJ datoteku treba ugraditi i podatke o tekstu programa. Prilikom linkovanja ne treba koristiti nikakvu posebnu opciju. Ako ovako pripremljenu aplikaciju startujete normalno osim malo veće veličine EXE datoteke nećete primetiti nikakvu raziku. De bi se startovao dibager treba otkucati **CLD <NazivPrograma>**. Posle ove komande se pojavljuje radni ekran dibagera odakle možete birati razne komande (slika 3.) Najednostavnije je da komanda birate iz opadajućih menija kombinacijom **<Alt-"X">** gde je "X" početno slovo u ispisanim opcijama. Na primer, pritiskom na **<Alt-F>** aktivira se **File** meni. Dalji izbor se izvodi kursorskim tasterima ili aktiviranjem posebno označenog tastera. Sva komande se mogu aktivirati i direktnim kucanjem u komandnoj liniji ili samo nekim tasterom. Na primer, umesto izbora opcije **Exit** iz menije **File** može se otkucati **Q<Enter>** u komandnoj liniji ili samo **<Alt-x>**. Ovde čamo opisati standardan postupak koji se najčešće koristi za detalje možete konsultovati dokumentaciju programa.

Obično se greška uočava u nekoj proceduri. Prvo što treba da uradimo po ulasku u dibager je da otvorimo tekst datoteke se tom procedurom. To se radi izborom opcije **Open** iz menije **File**. Posle toga treba uneti naziv datoteke u kojoj se nalazi kritična procedura. Zatim se kursorom dolazi na mesto pre kritične pozicije u programu – obično nekoliko naredbi ispred. Pritiskom na taster **<F9>** postavlje se prekidna tačka na naredbu u kojoj je kursor. Ovak kad bude tradalo de se izvrši označena naredba izvršavanje programe će se prekinuti i kontrola će se prepustiti dibageru. Zatim aktiviramo program komandom **<F5>**. Program će normalno radit sve dok na dode do naše prekidne tačke. Ukoliko prekidna tačka nije postavljena, program će normalno prekinuti rad. Ako želimo da aktiviramo dibager, dovoljno je da otkucamo **<Alt-D>** – program se trenutno zustavlje i kontrola se prepušta dibageru.

Posle toga možemo naredbama za izvršavanje programa korak po korak videti kako program zaista radi i kako manje sitnje promenljivih. Umesto tastera

<F9> mogli smo da pritisnemo taster <F7> on su tometski postavlja prekidnu tačku startuje program s tim što se po zaustavljanju prekidna tačka briše Komanda <F7> se zove "izvršavanje programa do kursora"

Za izvršavanje programa korak po korak na raspolaganju su komande **Step** i **Trace** Komandom **Step** (taster <F8>) program se izvršava naredbu po naredbu Taster <F10> ima slično dejstvo samo što izvršave procedure i funkcije kao celinu tj ne "ulazi" u njih Kucanjem komande **? <izraz>** računa se vrednost navedenog izraza Kucanjem **?? <promenljive>** možemo i menjati vrednost promenljive ili niza Prilikom na taster <F8> ispisuju se podaci o aktivnim radnim zonama otvorenim datotekama, vrednostima tekućih slogova i sl

Iz dibagera se izlazi pritiskom na taster <Alt-x>

RMAKE

Clipper se skoro nikad ne uči da bi se nap sao samo jeden program Obično se piše više programa i to prilično obima po broju linija U takvom radu veoma je važno da poštujete neke principe koje morate sami definisati

Preporučljivo je da za sve vaše programa izdvojite neke zajedničke procedure i njih grupišete u jednu celinu (najbolje bibl oteku) iz koje ćete po potrebi pozivati potrebne delove Ako ne postupite ovako vrlo brzo će vam se desiti da iste izmene pravite na nekoliko različitih mesta što je jedan od nekoliko najgorih poslova na svetu

Razvoj aplikacije u sebi sadrži i banalne greške tipa "de li sam preveo ovaj program pre linkovenja?" Mnogi ovaj problem rešavaju tako što za prevođenje svekog programa imaju BAT procedure koje uvek i znova prevode i linkuju celu aplikeciju

Već odavno svaki paket za razvoj progreme sadrži u sebi i jedan mal program po imenu *MAKE* koji progremeri zečudujuće retko koriste Taj mali programčić jednostavnim metodom rešava ogromne probleme koji mogu nastati pri izradi aplikecije Uz *Clipper* se isporučuje program po imenu *RMAKE* Šta je to u stvari? Ovekvi programi služe za kreiranje "intelligentnih" BAT procedure Program na osnovu opisa module vaše aplikacije i vremena kreiranja PRG i odgovarajućih OBJ datoteke automatski prevodi potrebne module i daje nepogrešivo korektnu EXE datoteku

Da bi se koristio jedna *MAKE* program potrebno je pre svega napraviti opis aplikacije U slučaju *RMAKE* o to treba de bude datoteka čstog mena kao i glavni program (preporučljivo) e ekstenzije RMK U toj datoteci se navode razne komande *RMAKE* programu e najvaznije je komenda za specifičavanje zavisnosti Na primer ako u neš program ulazi I modul po imenu **PROBA** tekst programa će se nalaziti u datoteci **PROBA PRG** e iz te datoteke se prevođenjem dobije datoteka **PROBA OBJ** Posievlje se pitenje kada treba prevoditi datoteku **PROBA PRG**? Pa uvek kad se u njoj vrše neke izmene A to se može preporučiti po tome što će tada vreme modifikacije datoteke **PROBA.PRG** biti "veće" od vremena modifikacije (kreiranje) datoteke **PROBA OBJ**! Ne ovom jednostavnom principu *RMAKE* i radi U RMK datoteci je potrebno somo navesti de se datoteke **PROBA OBJ** dobija" od datoteke **PROBA PRG** i treba navesti komende za jo dobijanje To se postiže sledećom naredbom

```
proba obj : proba prg
   clipper proba /n
```

U prvom redu je navedeno da datoteka **PROBA.OBJ** zavisi od datoteke **PROBA PRG** i da uvek kad je **PROBA PRG** menjene posle **PROBA OBJ** treba izvršiti komandu za dobijenje datoteke **PROBA.OBJ** navedenu u sledećem redu **clipper proba /n** Datoteka može zavisiti od više datoteka Na primer

```
proba obj . proba prg common-ch ui prg
   clipper probe
```

Zračenje navedene naredbe je uvek kada je bar **jedne** od datoteka navedenih u sp sku sa desne strane menjana Izmen /kreiraj i datoteku sa desne strane Ovekve komande se ne moreju sastojati samo od zavisnosti za OBJ i PRG datoteke na ‘sti način se može definisati zavisnost i za sve ostale tipove datoteke Na primer za EXE datoteku zavisnost se može ovako definisati

```
demo exe : demo obj ui.obj utl obj
@t2 lib deqo rmk
   rtlink file demo,ui,utl lib at2
```

Smisao navedene definicije je da uvek pošto se izmeni jedna od datoteka sa spiska treba iznova i nkovati program Primetite de se u spisku nalaze i neke datoteke koje ne ulaze stvarno u EXE datoteku biblioteka **NT2 LIB** i *make* opis **DEMO RMK** međutim uvek kad se oni manjaju takode treba kreirati novu EXE datoteku

Cele *make* datoteka se sastoji od ovakvih naredbi Ne primar za neku aplikaciju koje se sastoji od module **DEMO UI i UTL**, *make* daloteka može biti

```
demo obj   demo prg
   clipper demo /m /w

ui obj   ui prg
   clipper ui /n /w

utl obj : utl prg
   clipper utl /n /w

demo-exe   demo obj ui obj utl obj de
@o rmk
   rtlink /pll base50 file de
mo,ui,utl
```

Posle kra renja ovakve datoteke *možete* bez ra zmišljanja razvijati aplikaciju – jednostavno kucanje **RMAKE DEMO** će uredili sve potrebne operacije da bi se kreirale korektna EXE deloteka Već jedini zadatak je da svaki novi modul aplikecije uredno upišete u *make* doteloku i time ste završili posao

Ovim se funkcija *make* sistema nikako ne završave *može* sa još puno loga uradili za lakše pisanje programe Prvo primetite de se sve OBJ datoteke od PRG deloteke dobijeju ne isti nečin Nepotrebno je u vek iz početke nevodili komandu **CLIPPER** – treba saopštiti *make* programu keko se od PRG datoteka dob ju odgovarajuće OBJ deloteke Za to se koristi sledeća direktive

```
prg obj
   clipper $* /n
```


Pri tome je oznake $* specijelni makro označava konkretno ime koje će imati ekstenziju **PRG** Posle navođenja ove komende ne treba navod ti komandu kad god je u pitanju odnos PRG OBJ datoteka čime se puno dobija na preglednosti *make* datoteke

make program sadrži i standardni preprocesor sa direktivama **#include #ifdef** itd koje sa korisno mogu upotrebiti Neveščiju jedan čest primer Kada pišete jedno aplikaciju obično postoje tri različita slučaja prevođenja/linkovanje

- Radna verzija Tada sve module treba prevesti se opcijom /B da bi se mogao koristiti dibager a linkovanje treba izvršiti sa opcijom /PLL i /INCREMENTAL zbog ubrzanja linkovanja
- Verzije za isporučivanje Sve module treba prevesti tako da se zbace informacije za dibager i o brojevima i nije Linkovanje treba izvršiti bez opcije /PLL i /INCREMENTAL
- Demo verzija Ova verzija je ista kao verzije za isporučivanje samo što prevođenje treba vršiti se opcijom /Ddemo kako bi se u aplikaciju ugradile naredbe koje ograničavaju upotrebu demo programa.

Rešenja ovih problema mogu biti različita i uglavnom zavise od konkretnih uslova Je obično koristim sledeću *make* datoteku

```
#
#  RMAKE file
#

# Radna verzija
prg obj :
clipper $* /m /n /b

lnk   RTLink /PLL base50 @file lnk

# Demo verzija
#ifdef demo
! del *.obj
lnk = RTLink @file lnk
prg obj
clipper $* /m /n /l /Ddemo
#endif

# Exe verzija
#ifdef exe
#!del *.obj
lnk   RTLink @file lnk
prg obj
clipper $* /m /n /l
#endif

demo-obj   demo prg

ui obj   ui prg

utl obj   utl.prg

demo exe   demo obj ui obj utl obj de
mo rmk
   $(lnk)
```

Nerevno deo za kontrolu verzije se ne more posebno pisati u svakoj *make* datoteci već se može u ključivati po potrebi direktivom **#include**

U prethodnom primeru je upotrebljena i mogućnost direktnog zadavanje DOS komendi posle **#!** se može napisati DOS komande koje će biti izvršene bez ikakve provere uslove U ovom primeru se to koristi za brisanje svih OBJ datoteka kako bi se dobila korektna demo ili izvršna verzija

RMAKE poznaje i standerdne makros približan e kvivelent promenljivih u programskim jezicime Makro promenljivoj se može dodeliti vrednost naredbom dodeljivanja ne primer **lnk = rtlink @file lnk** Makroj se zatim mogu razvijati uz pomoć operatora **$** Na primer naredba **$(lnk)** ima isto dejstvo kao da smo napisali **rtlink @file lnk**. Ove mogućnost se ob čno koristi za skraćenje p sanja. Drugi način kreiranje makros je iz komadne l nije opcijom **/D<naziv makro promenljive>** Ne primer *make* se može startovati sa **RMAKE DEMO /Deze** čime se kreire mekro promenljiva **EXE** Direktive **#ifdef** služe za ispitivanje da li makro postoji tako vršimo grenanja u *make* programu U prethodnom primeru ako *make* startujemo sa opcijom **/Ddemo** kreiraće se makro **DEMO** izvršće se deo posle **#ifdef demo** čime će se dobiti specifikacija za demo verziju programa

RUN TIME OKRUŽENJA

Najosetljiviji deo razvoja programa je prva upotreba nekoliko prolnih podataka se tada zamenjenuju sa ogromnim količinama stvarnih podataka utrošak memonje rapidno raste i program se postavlje u jedno vrlo nestabilno stanje ako ste negde zeboravili da zatvorite otvor – greške će neumitno procuriti Zato pravilen red programe treba planirati u detalje jer kasnije greške mogu rezultovati gubitkom velikih količina podataka

Osnovni problem koji se ranije javljeo kod programa pisanih u starijim verzijame *Clipper* e je nedostatak memonje Osim standerdna upotrebe memorije za kod i promenljive zbog ubrzanje rade neophodno je u memoriji držati i dar delove indeksn h i DBF datoteke Narevno ove datoteke mogu bit **ogromne** pa uglavnom nije moguce držati celu datoteku u memoriji eli nezavisno "baterisanje" je svakako preko potrebno Ako se ne sve te memorijske zehteve doda i velki kod za pristup bazi podataka stringovi i memo polja velika dužine brzo se dolazi do veome skromne granice od 600K. Upotreba EMS memorije za neke potreba (indeksi) je korak napred ali sa druge strane kod nezaživo napreduje tako da su sve ta rešenja primenjena u starijim verzijama bile sasvim polovične i nategnute" Jedino rešenje su bil overleji eli je red sa nimce bio zaista mučan program je trebalo cepkati na male će stoš neprirodno organizovane delice moralo se voditi računa o tome iz kog overleja sa čita pozivaš slično

Clipper 5 0 je rešio u potpunosti problem memorije uvođenjem nekoliko novih sistema za upravljenje memorijom Njihov rad je potpuno automatski i programer neme ni kakve potreda de upravlja njihovim radom (sa izuzetkom podešavanja raznih parametare) ali je potrebno znati kako sve to radi de bi program maksimalno prilagodili i ubrzali

Kada se program linkuje *rTlink* kreira dinamičke overleje sve *Clipper* procedure i funkcije deli u **stranice** koje upaluje na krej EXE datoteke ili u posebnu overlej deloteku U jednu stranicu upisuje se više jedna

ili samo deo procedure, u zavisnosti od veličine. Precedure pisane u drugim jezicima ne ulaze u dinamičke overleje.

Po startovanju programa, prvo se alocira memorija za deo programa koji je stalno u memoriji. Tu spadaju *sistemske* procedura za prestup datotekama, *ulaz/izlaz na ekran*, itd. Ovom delu se prikljućuju i delovi programa pisani na C-u i asembleru. Veličina ovog prostora se određuje za vreme linkovanja programa i *RTlink* je uvek saopštava po kreiranju EXE datoteke u zagradi.

Pošto se učita inicijalni deo aplikacije vrši se rezervacija fiksnog dela memorije. To je deo koji sadrži razne sistemske tabele i fiksno alociranu memoriju. Na primer, ako u C-u alocirate memoriju, to će povećati ovu zonu u memoriji. Veličina ove zone se kreće od 100 – 200K.

Na kraju se sva preostala memorija (ukoliko nije upotrebljen parametar X za smanjenje raspoložive memorije) dodeljuje menaxeru virtuelne memorije (VMM). To je vrlo važan sistem preko koga se odvija kompletna "komunikacija" programa i memorije. Kada programu zatreba memorije, upućuje sa zahtev VMM-u. Ovaj odvaja jedan ili više blokova (segmenata) potrebne memorije i vraća kontrolu glavnom programu koji dalje može koristiti alocirane segmente. Sve se to dešava u osnovnih 640K memorije. Međutim šta ako sa desi da program zatraži memoriju a na postoji slobodni prostor za alociranje? Najjednostavnije rešenje je da sistem ne upravljanje memorijom tada javi grešku i prekine program. Ali VMM se tu na "predaje" ako raspolaže sa EMS memorijo, nekoliko segmenata koji su u osnovnih memoriji prebacuju se privremeno u EMS memoriju, kako bi se oslobodila memorija za nove segmente. Problem je koje segmente premestiti u EMS memoriju, jer ako program ponovo zatraži segmente koji su upravo izbačeni VMM će morati ponovo da ih učitava iz EMS memorije i da bi njih učitao moraće neke stanje da izbaci. Pošto ovakva operacija traje izvesno vreme, bilo bi dobro da se maksimalno smanji broj prebacivanja. Postoje standardni algoritmi koji se primenjuju u ovakvim slučajevima a u *Clipper*-u se primenjuje **LRU** (*Least Recently Used*) algoritam po kome sa uvek izbacuju segmenti koji su se redje koristili. Naravno, uvek postoji slučaj u kome svaki algoritam najgore radi, ali u većini praktičnih slučajeva ovaj algoritam sasvim dobro funkcioniše. Ukoliko sistem nema EMS memoriju ili je VMM već alocirao svu raspoloživu, za privremeno odlaganje blokova koristi se disk. To je dosta sporo ali, u slučaju nedostatka RAM memorije jedino rešenje. Na ovaj način VMM stvara privid da je memorija mnogo veća od raspoložive, čime se jedino malo gubi na brzini. Adresni prostor koji se "stvara" ovakvom tehnikom naziva se **virtuelni**. VMM podržava teoriju virtuelni adresni prostor od 64MB, što je dovoljno i za najveće apetite za memorijom. Za realizovanje virtuelnog adresnog prostora otočno se koristi tzv *protected mode* procesora 80286 80386 i dalje dok 8086 nema takvu mogućnost. *Clipper* ne koristi ovakav (hardverski) metod već se sve izvodi softverski, čime je omogućeno da programi rade i na 8086 procesorima. Naravno performanse su tada daleko lošije ali program ipak radi. Treba napomenuti da je prisustvo EMS memorija u sistemu neophodno za kvalitetan rad VMM-a on će raditi i samo sa diskom, ali daleko sporije.

DINAMIČKI OVERLEJI

VMM je osnovni sistem za upravljanje memorijom na koga se oslanjaju svi podsistemi koji koriste memoriju. Jedan od tih podsistema je menaxer dinamičkih overleja. Već je rečeno da se pri linkovanju aplikacija deli na strane jednake veličine. Kada se program startuje menaxer učitava potrebne dinamičke strane u virtuelnu memoriju. Kako program radi, po potrebi se učitavaju nove strane sa diska. Pošto VMM koristi LRU algoritam najčešće koričćeni blokovi se poste kraćeg rada prebacuju, ili u osnovnu ili u EMS memoriju. Neravno blokovi virtuelne memorije koji sadrže dinamičke strane EXE datoteke neće se upisivati na disk u nedostatku EMS i osnovne memorije: one se jednostavno brišu iz memorije a po potrebi se ponovo učitavaju iz overlej datoteke.

U praksi dinamički overlei rade savršeno. U početku se uočava blage vremenska pauza dok su sve dinamičke strane "ravnopravne", ali se vrlo brzo rad u brzava jer se najčešće koričćene strane premeštaju u osnovnu i EMS memoriju tako da pristup disku postaje redi. Što se tiče zauzeća memorije, dinamički overlei rešavaju većinu problema i najveće aplikacije imaju vrlo prihvatljive zahteve za memorijom. Na primer, EXE program veličine od preko 800K, može da radi sa samo 350-400K slobodne osnovne memorije!

Dinamički overlei se kreiraju automatski, bez učešća programera. Bez obzira što i dalje postoji mogućnost koričćenje statičkih overleja, zaista ne postoji potreba za njima. Pri pisanju programa treba imati u vidu da velika količina koda pisanog u *Clipper*-u vrlo malo utiče na rast programa. Najveći rast je kada se program linkuje sa procedurama u C-u ili asembleru – ove procedura se ne smeštaju u dinamičke overleje i povećavaju osnovni deo aplikacije. Naročito je spasno alocirati velike količine memorije iz takvih modula, jer se na taj način smanjuje memorija koju koristi VMM.

Sigurno ste primetili da je u *Clipper*-u moguće dinamički kreirati veoma duge nizove ili velike string promenljive bez prethodne deklaracije. Kada se u programu izvrši naredba poput:

```
x := Space(50000)
```

kreira se promenljiva **X** i njena vrednost (prostor od 50K ispunjen simbolom " "). Ako se sada izvrši naredba

```
x := 12
```

promenljiva **X** gubi prethodnu vrednost (i tip) i dobija novu vrednost 12. Međutim, sada ostaje nepotrebno alocirano 50K memorije! Ovakva situacija se javlja mnogo češće između ostalog pri kreiranju raznih međurezulatata i slično. Posebam podsistem se brine o ovakvim slučajevima. To je tzv **SVOS** (*Segmented Virtual Object Store*). On između ostalog smanjuje fragmentaciju virtuelne memorije nastalu brisanjem velikih količina podataka i vrši tzv "skupljanje đubreta" – ponovo skuplja raskorišćena delove za korišćenje. Ipak pravilo dobrog programiranja je da ne treba zloupotrebljavati mogućnost dinamičkog kreiranja nizove ako je nije potrebno, uvek je bolje niz deklarisati kao **STATIC** ili **LOCAL**, čime se znatno dobija na brzini

ORGANIZACIJA PROGRAMA

Kada ma kom programeru pomenete nešto o kvalitetu pisanje programe sasvim je sigurno da neće te naći niti jednog koji smatra da su njegovi programi loše pisani – svako smatra da je baš njegov način i jempo komentarisanje najbolji. Korisnički interfejs najlepi i ei. Ipak vrlo često osjeću nekoliko važnih činjenica koje ostaju skrivene. Programe pisane u *Clipper*-u treba (baš kao i sve druge programe) pravilno organizovati. Pod jim se naravno podrazumeve da se pri pisanju programa treba držati opštih pravila kao što je dosledno komentarisanje i "nazubljivanje" listinga, i sl. Ali i nešto više. Naime *Clipper* programi u odnosu na našo što bi uslovno mogli nazvati "standardni programi" imaju veoma poljačanu komunikaciju se korisnikom i imaju osobinu da "brzo rastu" – mnogi delovi programa se dobijaju jednostavnim kopiranjem nekog već postojećeg dela i njegovim preprevljenjem. Na stranu što je ovo vrlo loša tehnika, jer se one i tako koristi. Valjda zejo za *Clipper* postoji jako mnogo raznih generatora programa koji automatski generišu kostur ili cele aplikacije. Njihove prakične vrednost je posebna tema o kojoj se može diskutovati.

Bez obzira na raznovrsnost aplikacije koje pokriva *Clipper* možete očekivati da će vaša aplikacija brzo narasti. Zbog toga je vrlo potrebno da se dosledno držite sledećih nekoliko pravila

- Takst programe organizuje po log čkim celinama. Obično sa u jednu datoteku smeštaju procedure koje realizuju jednu stavku menija. To će vam sa urednim komentarima višestruko olakšati kasnije analiziranje. Ako koristije sjelićke overleje ovo će vam olakšati kod dijeljenje aplikacije
- Vodite evidenciju dejoteka koje koristi program. Svaku izmenu u datotekama **obavezno** upišite i u dokumentaciju. Ove je vrlo važno a često se zaboravlja je bezi padataka podaci su potjednake važni kao i program

Što se tiče korisničkog interfejsa, imaje na raspolaganju veliki izbor ugrađanih komendi i funkcije e možete koristiti i neku nedgradnju u vidu kompletnog sistema. Na najnižem nivou mozeje koristiti naredbu **MENU TO** i funkciju **AChoice** za realizaciju menije. Nešto što se vrlo često zaboravlja je **dosledinost** korsničkog interfejsa, bez obzira na tip. Na primer neophodno je da korisniku omogućite da se u bilo kom trenutku može predomisliti i vratiti se jedan nivo nazad. Ne primer ako je greškom izabrao meni se čuvanje podataka, mora se omogućiti da se prethodne odluke suspenduje pritiskom na neki taster, na primer, **<Esc>**. Pravilno bi bilo da taj isti taster služi za ulaz u bilo kog nivoa iz aplikacije – totalno je besmisleno da korisnik pamti 10 raznih tastera za ulaz iz 10 raznih nivoa! Sasvim slično, srodne operacije bi trebalo skiverati istim tasterom u raznim situacijama. Na primer, ako se slog briše tasterom F3, onda neka to važi za celu aplikaciju. Ovakva pravila će vam najbolje reći korisnici vaših programa, ali se skoro sve svodi na jedno sasvim jednostavno – doslednost.

Početnici se obično namuče se organizacijom procedura u programu. Trebe znati da je većinu problema najjednostavnije predstaviti stablom, u kome sveki čvor predstavlja jednu operaciju. Mnogi treba da omoguće "šetanje" kroz to stablo. Struktura se može skomplikovati dodavanjem nekih pogodnosti, ali obično sledeća struktura procedure svakog čvora stabla zadovoljava u potpunosti

```
PROC Node
   WHILE .T.
      // Izbor opcije
      Choice := Menu (Options)
      // Analiza pritisnutog tastera
      DO CASE
         // Prethodni nivo
         CASE Choice == 0 : EXIT
         // Prva opcija
         CASE Choice == 1 : Proc1 ()
         // Druga opcija
         CASE Choice == 2 : Proc2 ()
         // Ostale opcije
         ...
      END CASE
   END DO
RETURN
```

Sad procedura **PROC1** mogu realizovati isto obrade ali mogu biti ponovo "čvorovi" stabla aplikacije, tj. imati strukturu sličnu prethodnoj procedure.